Asynchronously Parallel Optimization Solver for Finding Multiple Minima

Jeffrey Larson Stefan M. Wild

Argonne National Laboratory

August 10, 2016

Problem setup

Want to find distinct, high-quality minima for the problem

$$\begin{aligned} &\underset{x \in \mathbb{R}^n}{\text{minimize}} \; f(x) \\ &\text{subject to: } x \in \mathcal{D}, \end{aligned}$$

when $\mathcal{D}$ is compact, c concurrent evaluations of f are possible, and relatively little is known about f a priori.

Problem setup

Want to find distinct, high-quality minima for the problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x)$$

$$\text{subject to: } x \in \mathcal{D},$$

when $\mathcal{D}$ is compact, c concurrent evaluations of f are possible, and relatively little is known about f a priori.

- Derivatives of f may or may not be available
- c is fairly small
- Evaluating f is expensive
- High-quality can be measured by more than the objective

Motivation

Motivation

Initial approaches

- Grid over the domain (easily parallelizable)

Initial approaches

- Grid over the domain (easily parallelizable)
- Random sampling (easily parallelizable)

Initial approaches

- Grid over the domain (easily parallelizable)
- Random sampling (easily parallelizable)
- Evolutionary Algorithms (many are parallelizable)

Initial approaches

- Grid over the domain (easily parallelizable)
- Random sampling (easily parallelizable)
- Evolutionary Algorithms (many are parallelizable)
  - Genetic Algorithm
  - Simulated Annealing
  - Particle Swarm
  - Ant Colony Optimization
  - Bee Colony Optimization
  - Cuckoo Search
  - Bacterial Colony Optimization
  - Grey Wolf Optimization
  - Firefly Optimization
  - Harmony Search
  - River Formation Dynamics

Global DFO

Theorem (Törn and Žilinskas, *Global Optimization*, 1989)

An algorithm converges to the global minimum of any continuous f on a domain $\mathcal{D}$ if and only if the algorithm generates iterates that are dense in $\mathcal{D}$.

Global DFO

Theorem (Törn and Žilinskas, *Global Optimization*, 1989)

An algorithm converges to the global minimum of any continuous f on a domain $\mathcal{D}$ if and only if the algorithm generates iterates that are dense in $\mathcal{D}$.

- Either assume additional properties about the problem
  - convex f
  - separable f
  - finite domain $\mathcal{D}$

Global DFO

Theorem (Törn and Žilinskas, *Global Optimization*, 1989)

An algorithm converges to the global minimum of any continuous f on a domain $\mathcal{D}$ if and only if the algorithm generates iterates that are dense in $\mathcal{D}$.

- Either assume additional properties about the problem
  - convex f
  - separable f
  - finite domain $\mathcal{D}$
- Or possibly wait a long time (or forever)

Global DFO

Theorem (Törn and Žilinskas, *Global Optimization*, 1989)

An algorithm converges to the global minimum of any continuous f on a domain $\mathcal{D}$ if and only if the algorithm generates iterates that are dense in $\mathcal{D}$.

- Either assume additional properties about the problem
  - convex f
  - separable f
  - finite domain $\mathcal{D}$
- Or possibly wait a long time (or forever)

The theory can be more than merely checking that a method generates iterates which are dense in the domain.

Global DFO

Theorem (Törn and Žilinskas, *Global Optimization*, 1989)

An algorithm converges to the global minimum of any continuous f on a domain $\mathcal{D}$ if and only if the algorithm generates iterates that are dense in $\mathcal{D}$.

- Either assume additional properties about the problem
  - convex f
  - separable f
  - finite domain $\mathcal{D}$
  - concurrent evaluations of f
- Or possibly wait a long time (or forever)

The theory can be more than merely checking that a method generates iterates which are dense in the domain.

Multistart Methods

An algorithm must trade-off between “refinement” and “exploration”.

Multistart Methods

An algorithm must trade-off between "refinement" and "exploration".

- Explore by random sampling from the domain $\mathcal{D}$
- Refine by using a local optimization run from some subset of points

Multistart Methods

An algorithm must trade-off between "refinement" and "exploration".

- Explore by random sampling from the domain $\mathcal{D}$
- Refine by using a local optimization run from some subset of points

Desire to find all minima but start only one run for each minimum

Multistart Methods

An algorithm must trade-off between "refinement" and "exploration".

- Explore by random sampling from the domain $\mathcal{D}$
- Refine by using a local optimization run from some subset of points

Desire to find all minima but start only one run for each minimum

+ Get to use (more developed) local optimization routines.
  - least-squares objectives, nonsmooth objectives, (un)relaxable constraints, and more
+ Increased opportunity for parallelism
  - objective, local solver, and global solver
- Can require many sequential evaluations for the local solver

Where to start?

MLSL:

$\hat{x} \in \mathcal{S}_k$

Where to start?

MLSL: (S2)–(S4)

$\hat{x} \in \mathcal{S}_k$

(S2) $\nexists x \in \mathcal{S}_k$ *with* $[\|\hat{x} - x\| \leq r_k$ *and* $f(x) < f(\hat{x})]$

(S3) $\hat{x}$ *has not started a local optimization run*

(S4) $\hat{x}$ *is at least* μ *from* $\partial\mathcal{D}$ *and* ν *from known local minima*

Where to start?

MLSL: (S2)–(S4)

$\hat{x} \in \mathcal{S}_k$

(S1) $\nexists x \in \mathcal{L}_k$ with $[\|\hat{x} - x\| \leq r_k$ and $f(x) < f(\hat{x})]$

(S2) $\nexists x \in \mathcal{S}_k$ *with* $[\|\hat{x} - x\| \leq r_k$ *and* $f(x) < f(\hat{x})]$

(S3) $\hat{x}$ *has not started a local optimization run*

(S4) $\hat{x}$ *is at least* μ *from* $\partial\mathcal{D}$ *and* ν *from known local minima*

APOSMM: (S1)–(S4), (L1)–(L6)

$\hat{x} \in \mathcal{L}_k$

(L1) $\nexists x \in \mathcal{L}_k$ $[\|\hat{x} - x\| \leq r_k$ and $f(x) < f(\hat{x})]$

(L2) $\nexists x \in \mathcal{S}_k$ with $[\|\hat{x} - x\| \leq r_k$ and $f(x) < f(\hat{x})]$

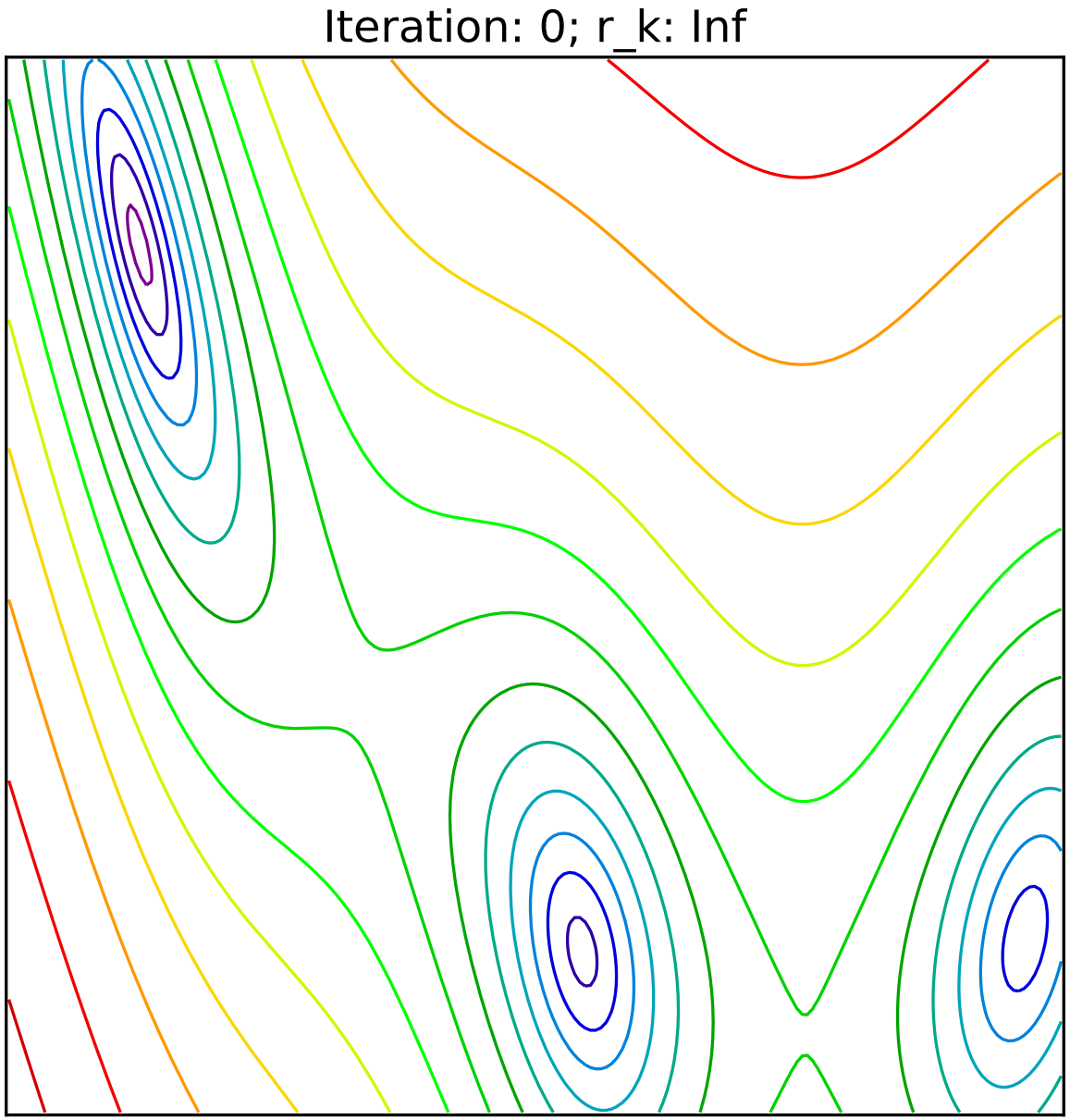
(L3) $\hat{x}$ has not started a local optimization run

(L4) $\hat{x}$ is at least μ from $\partial\mathcal{D}$ and ν from known local minima

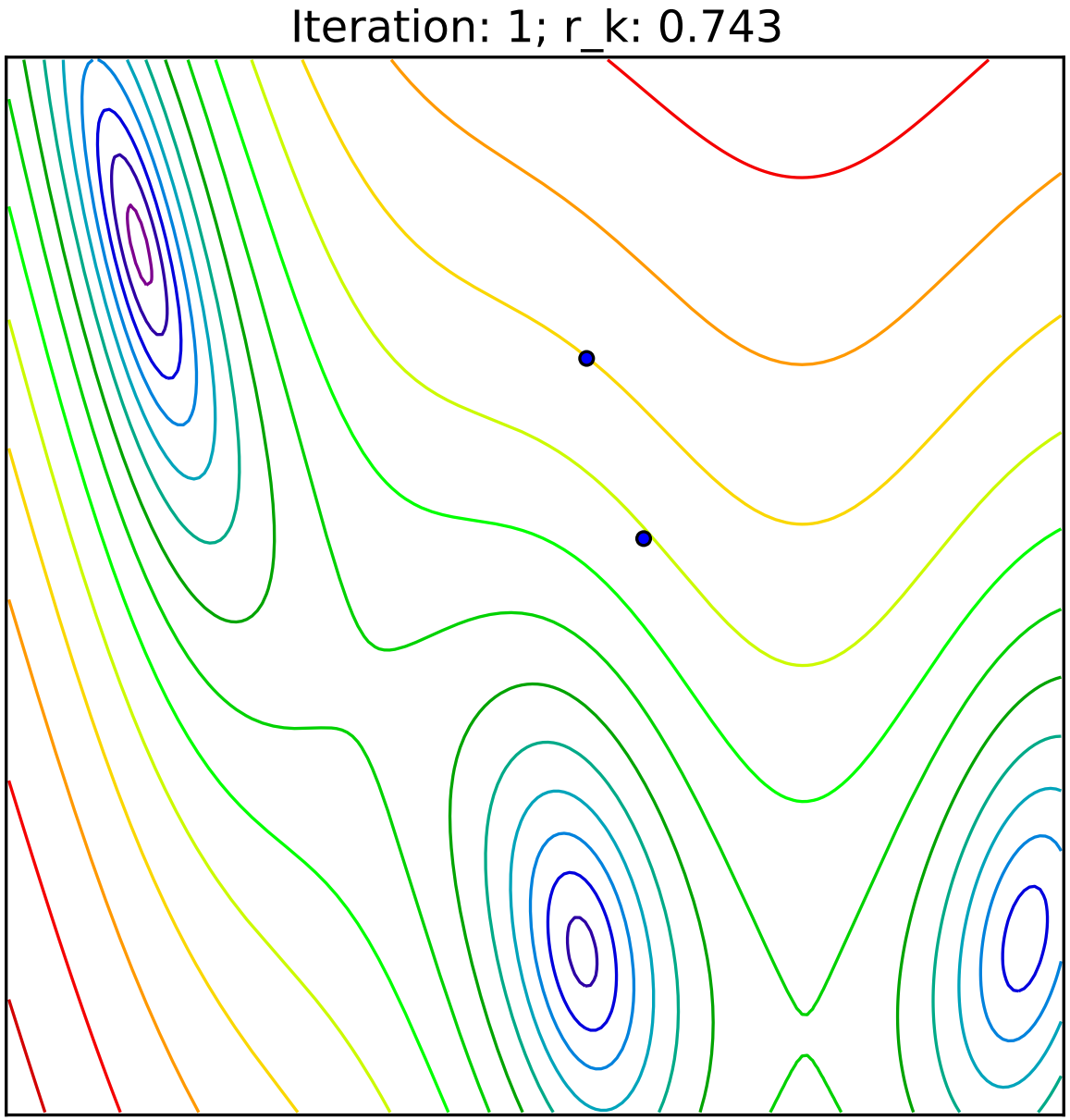
(L5) $\hat{x}$ is not in an active local optimization run and has not been ruled stationary

(L6) $\exists r_k$-descent path in $\mathcal{H}_k$ from some $x \in \mathcal{S}_k$ satisfying (S2-S4) to $\hat{x}$

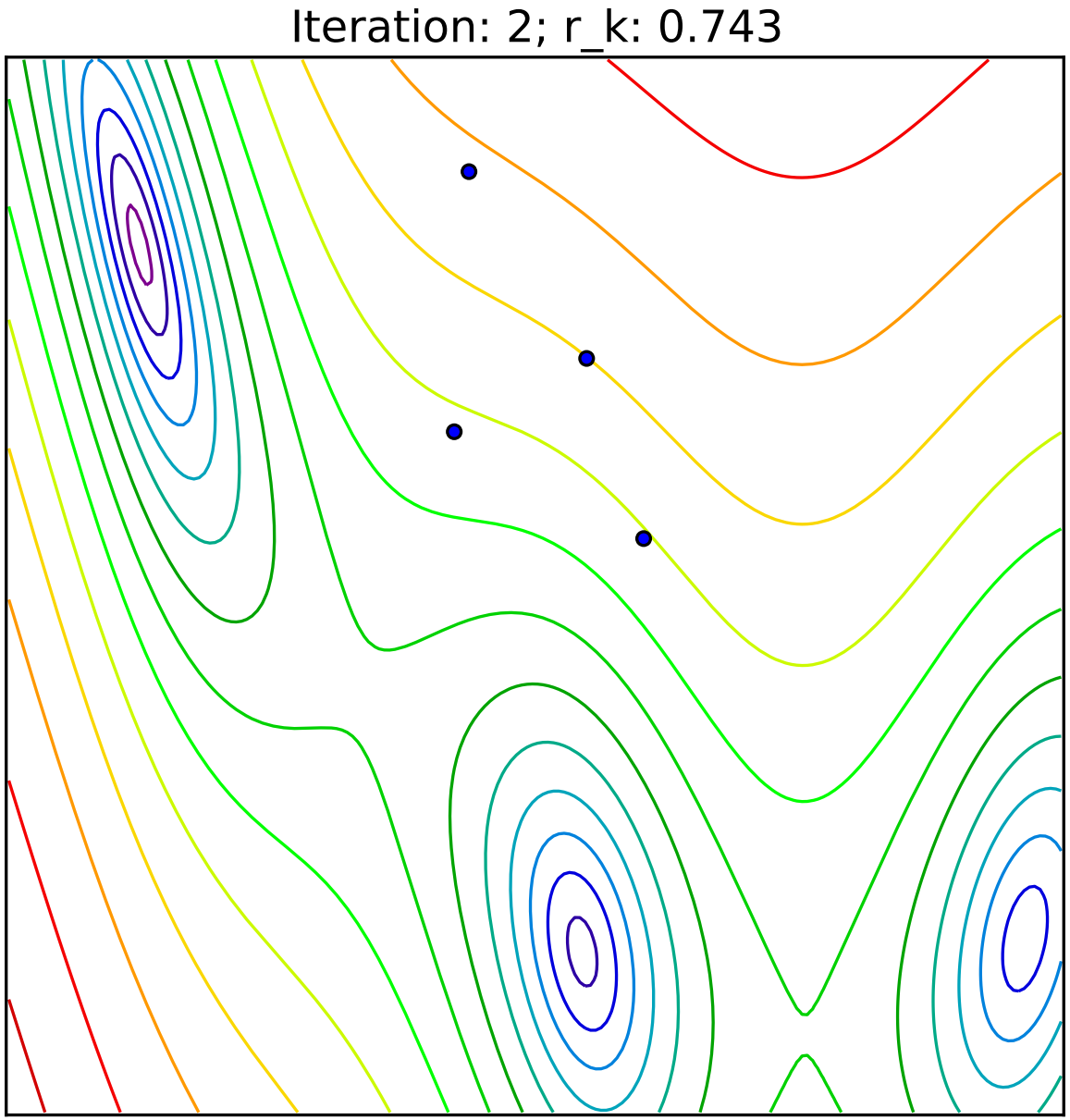
APOSMM



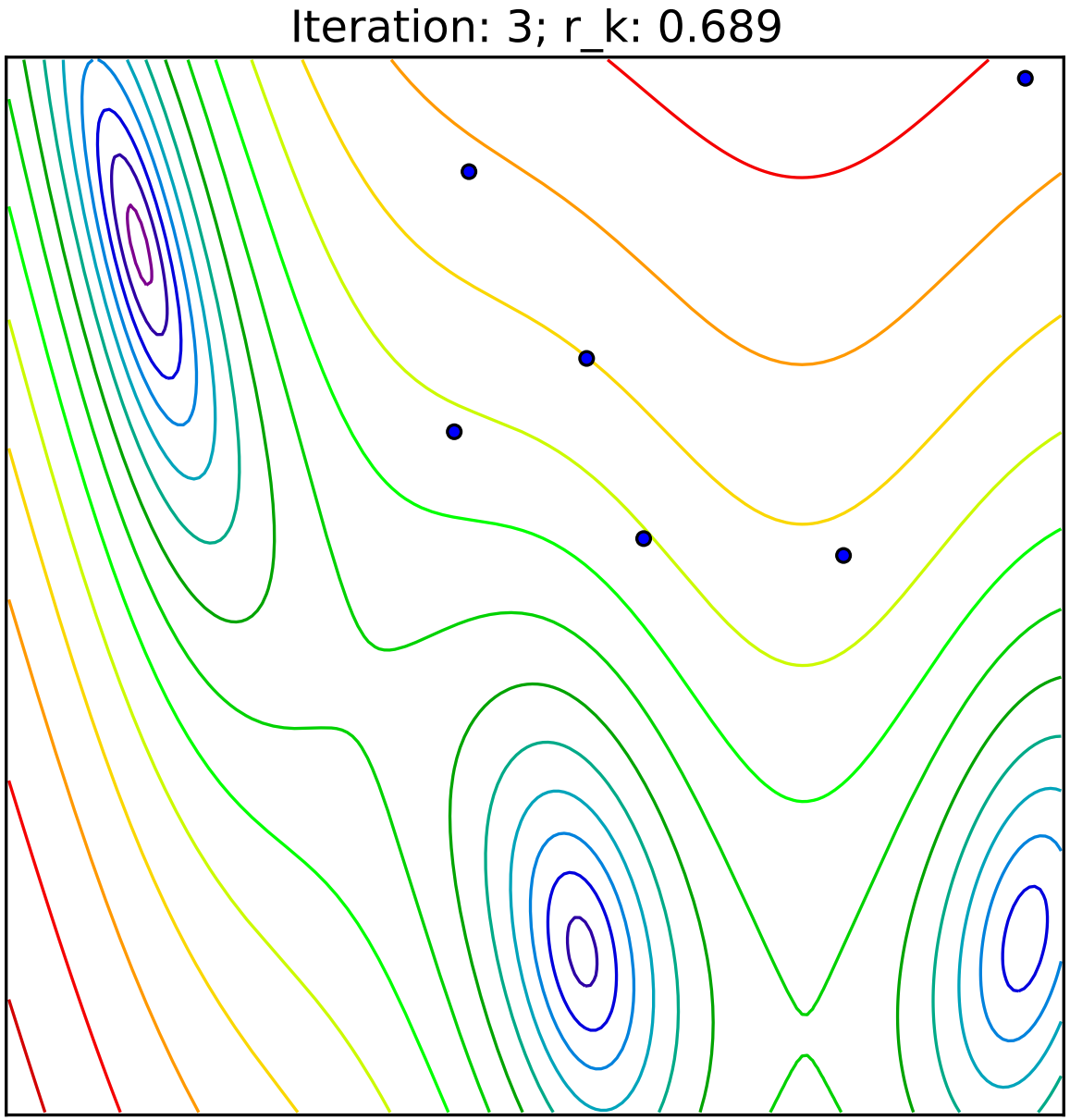
APOSMM



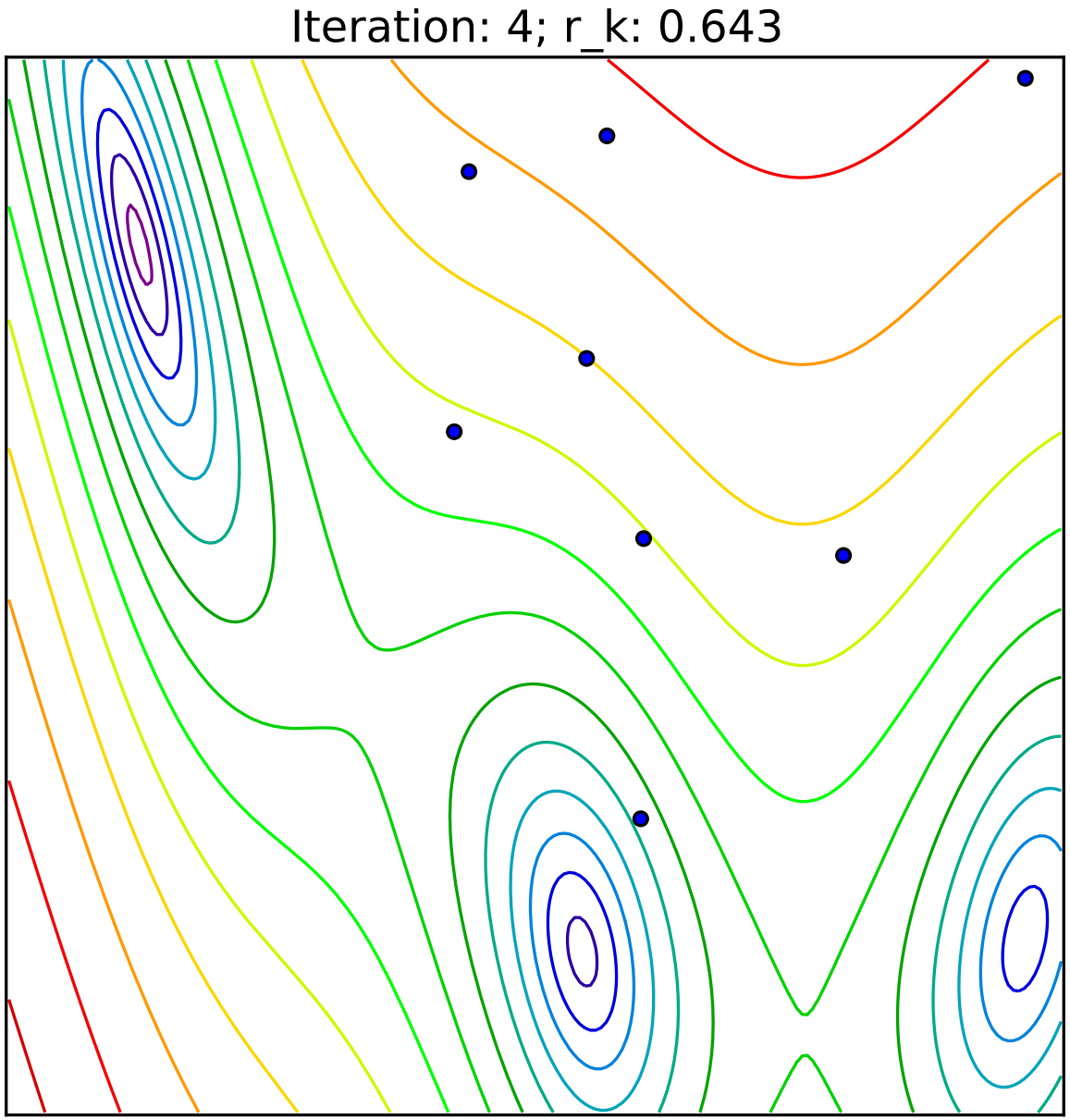
APOSMM



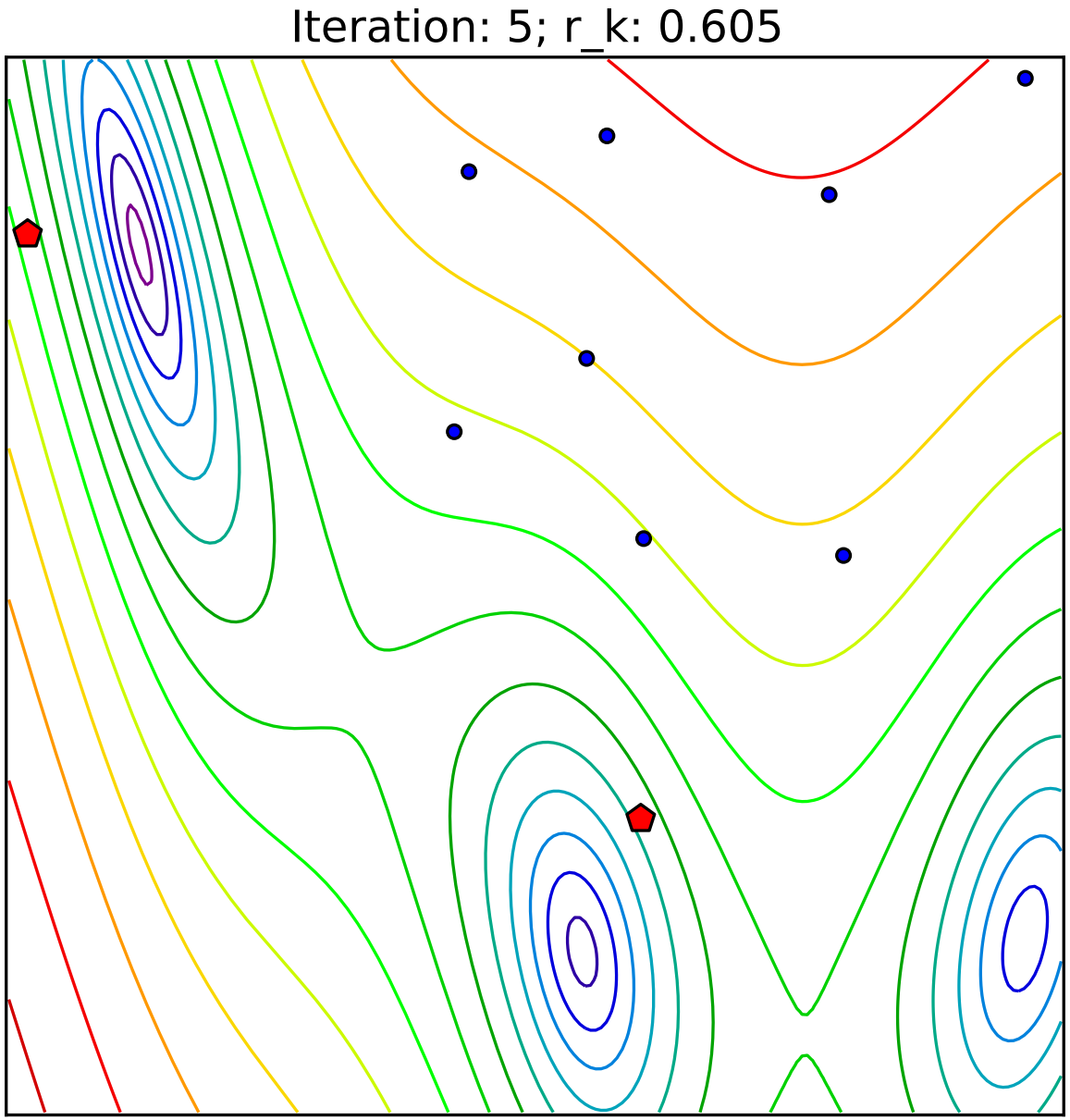
APOSMM



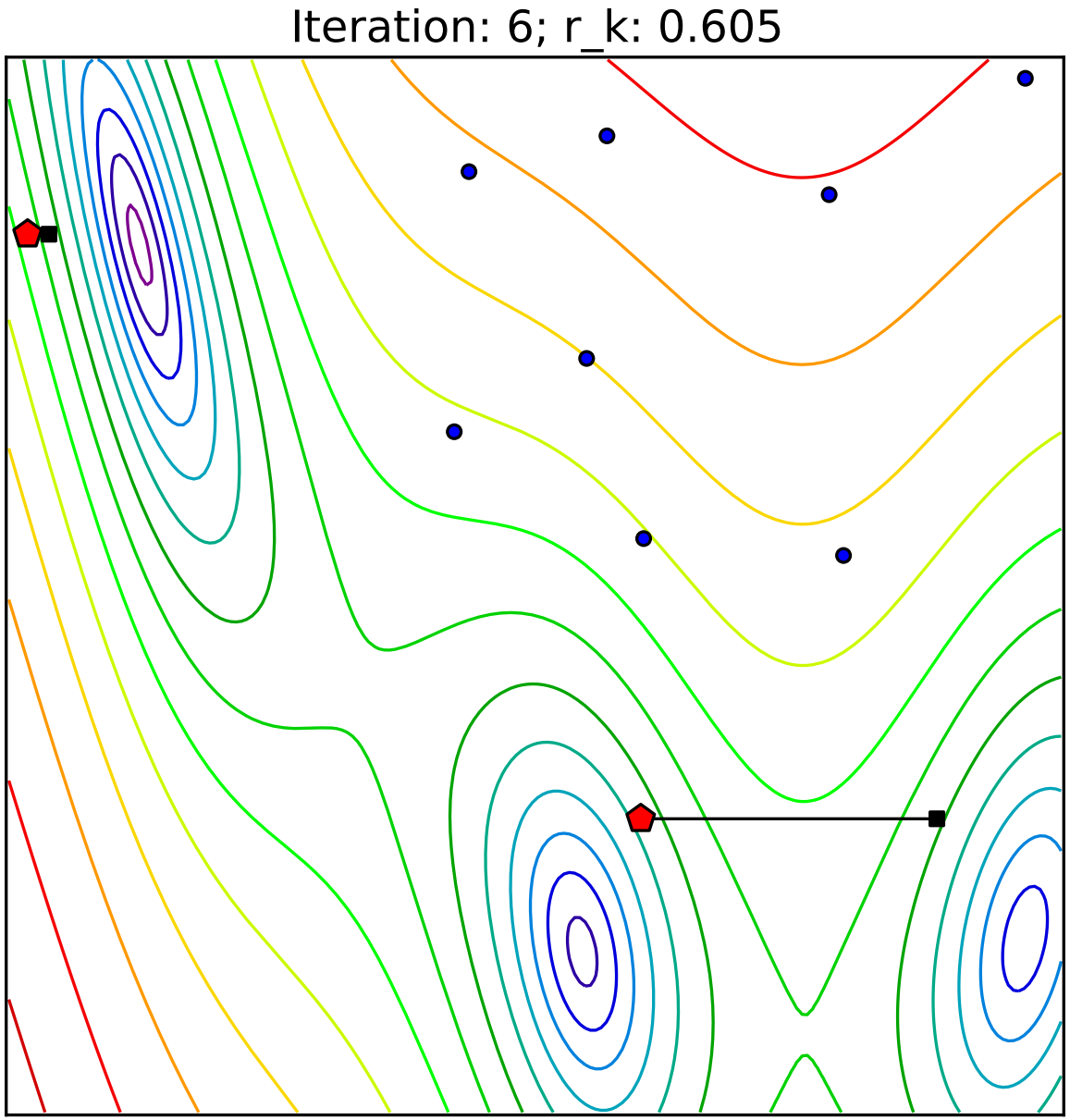
APOSMM



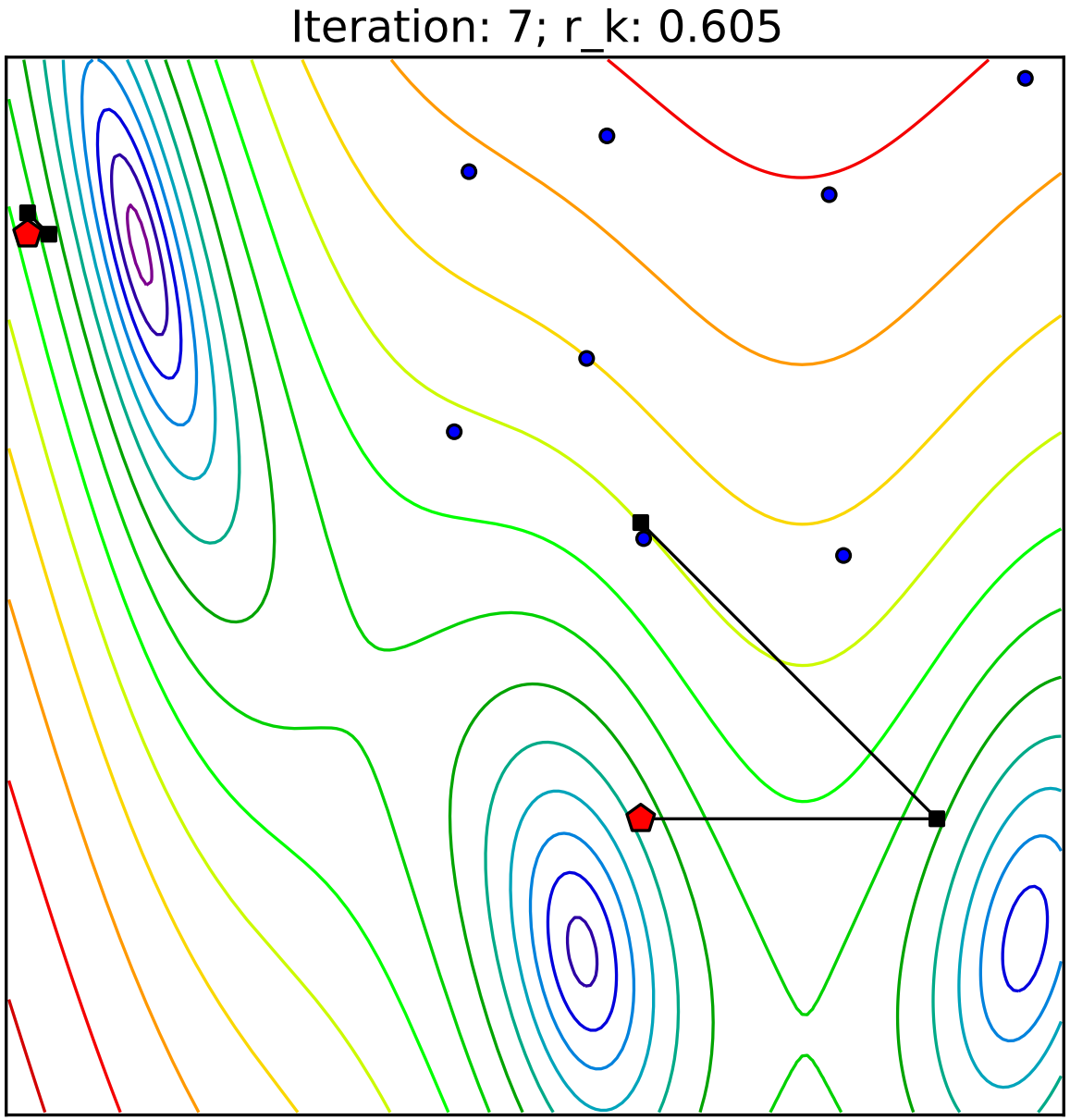
APOSMM



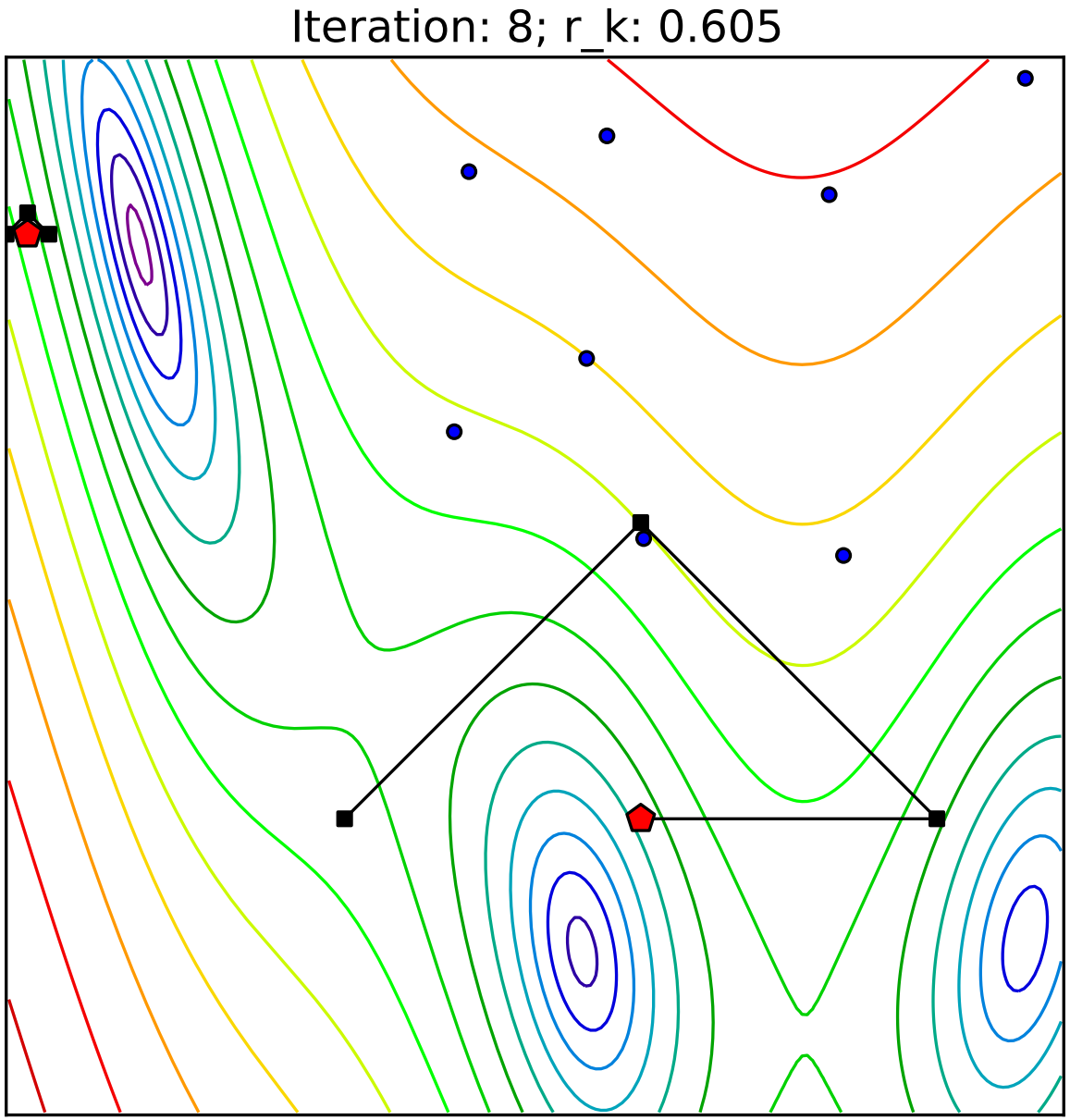
APOSMM



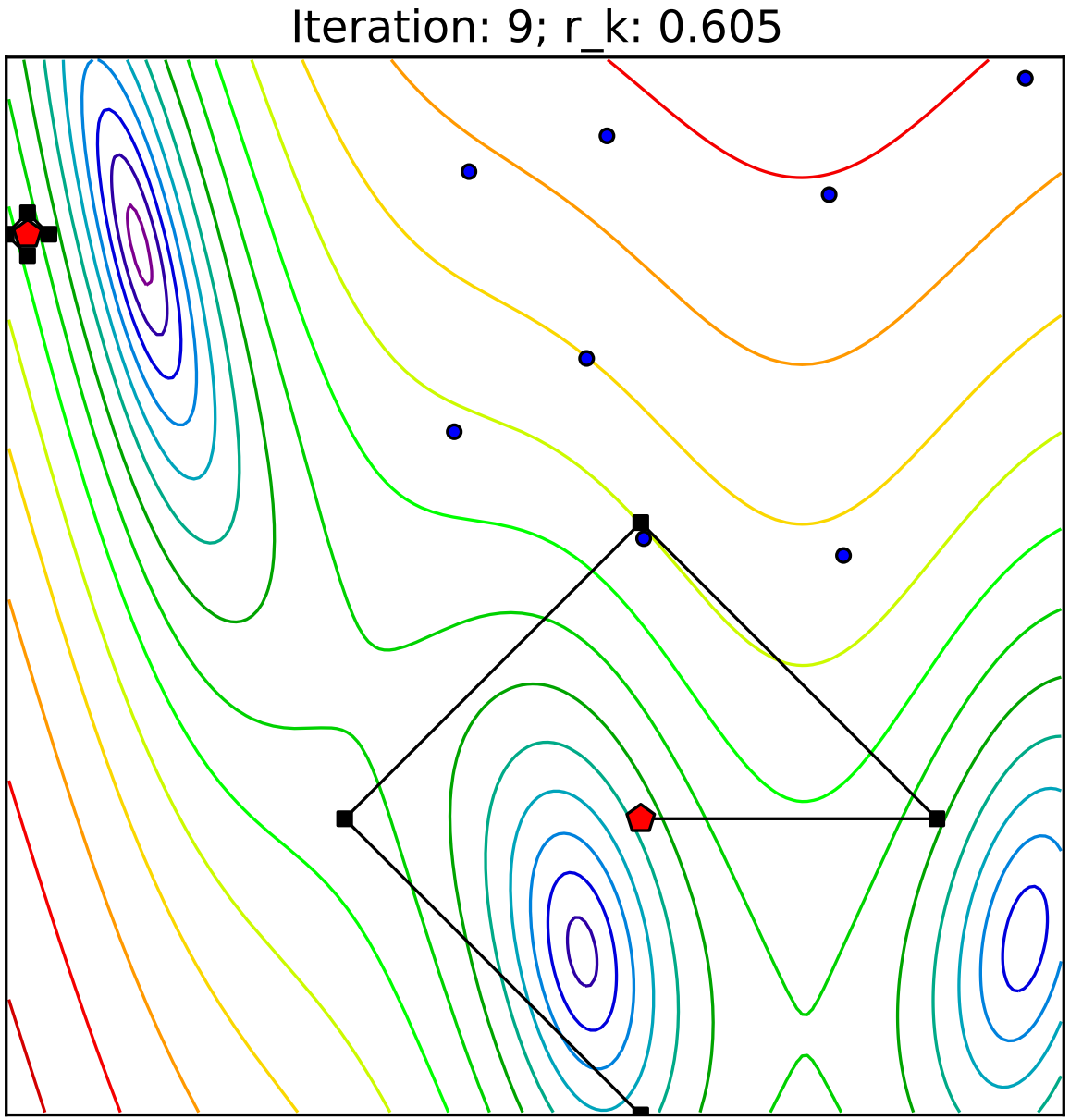
APOSMM



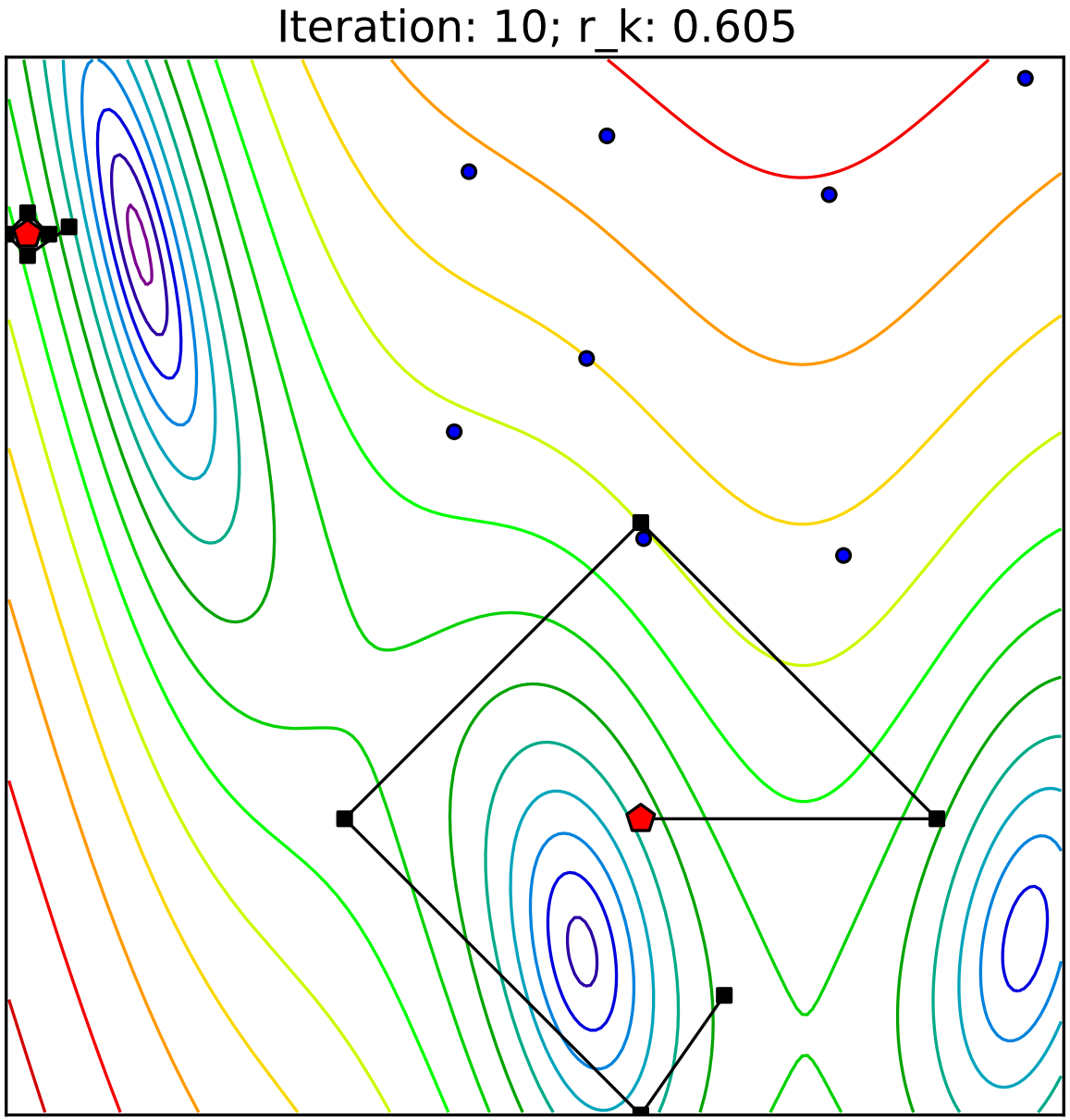
APOSMM



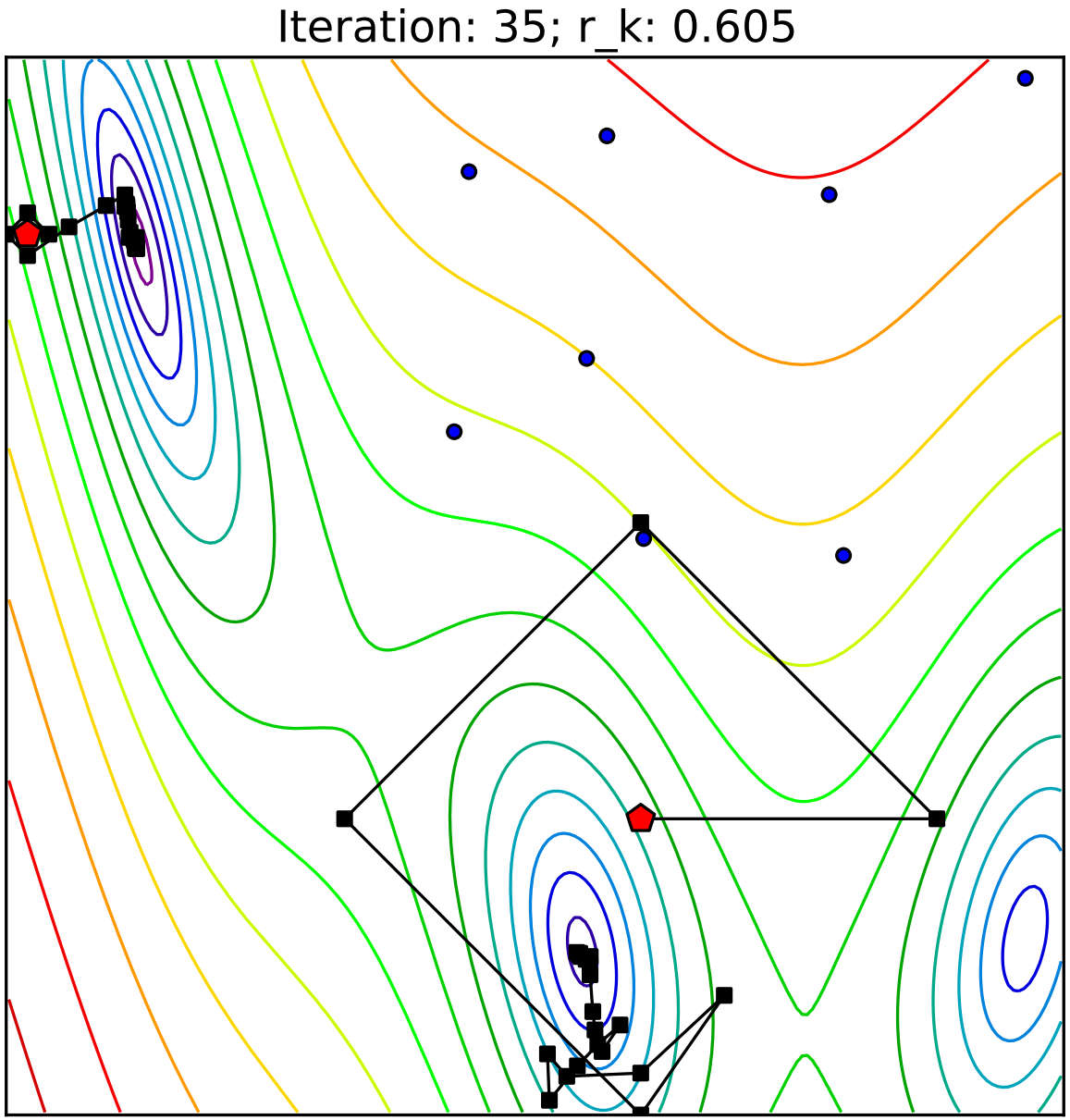
APOSMM



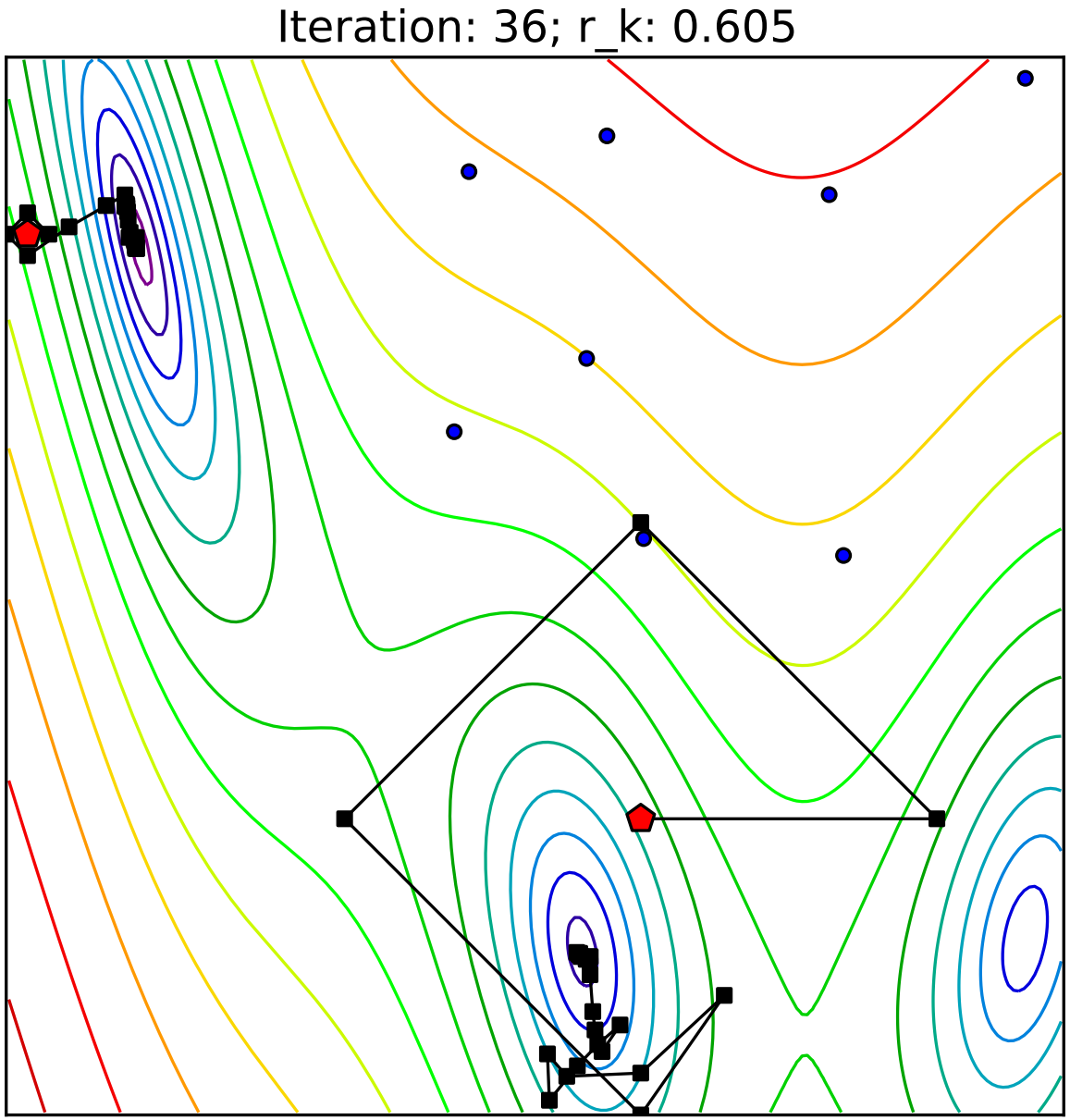
APOSMM



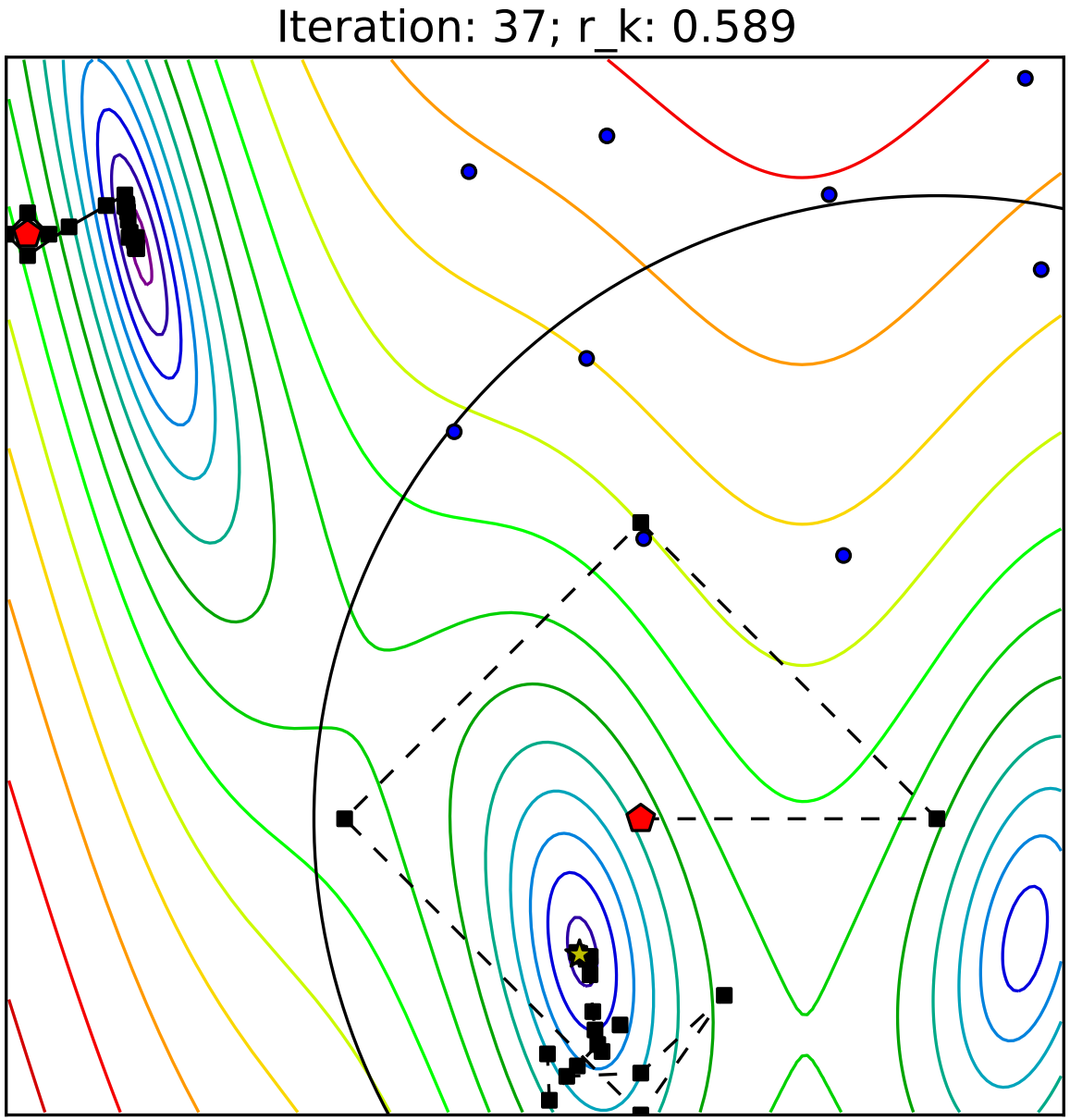
APOSMM



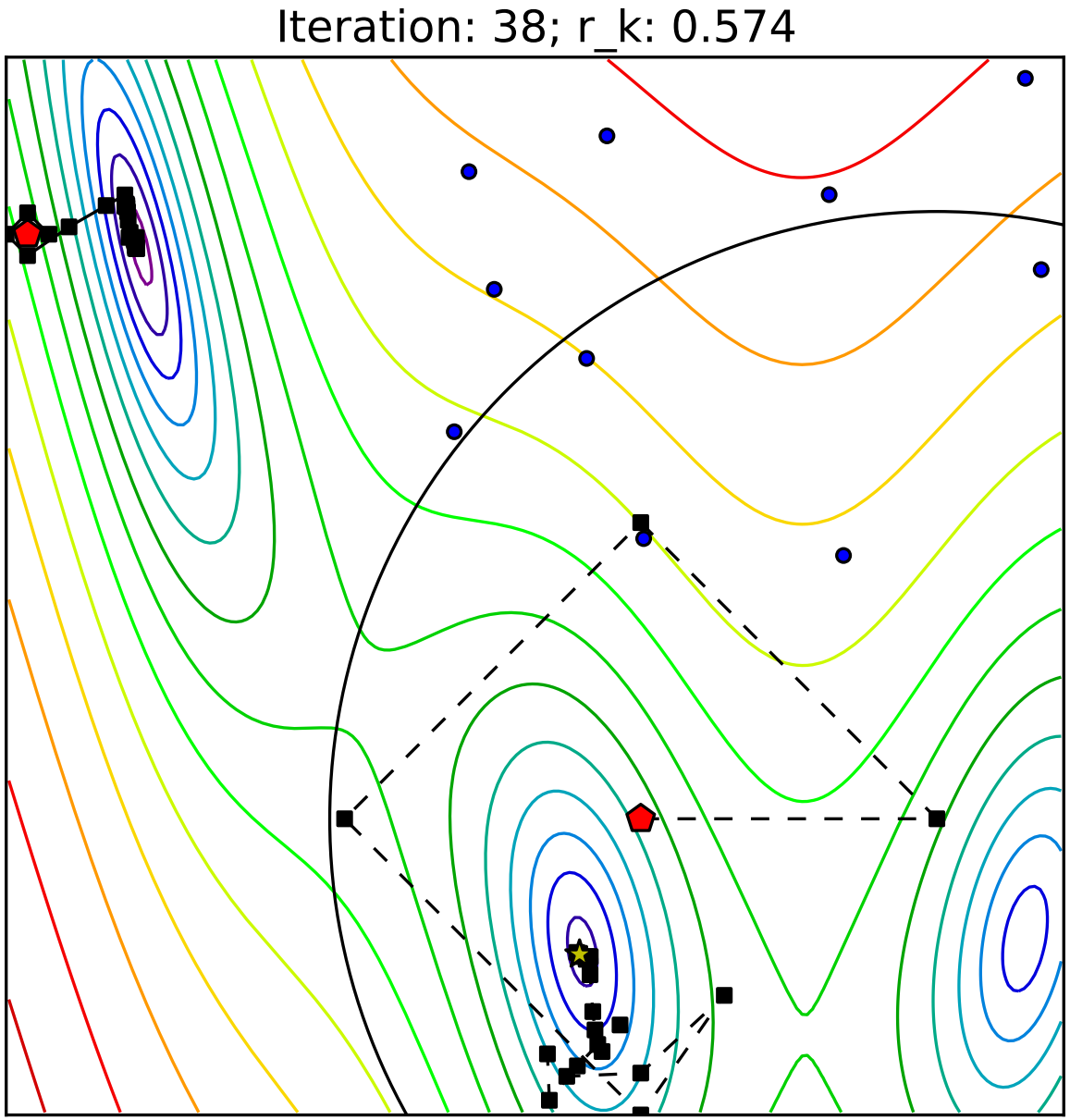
APOSMM



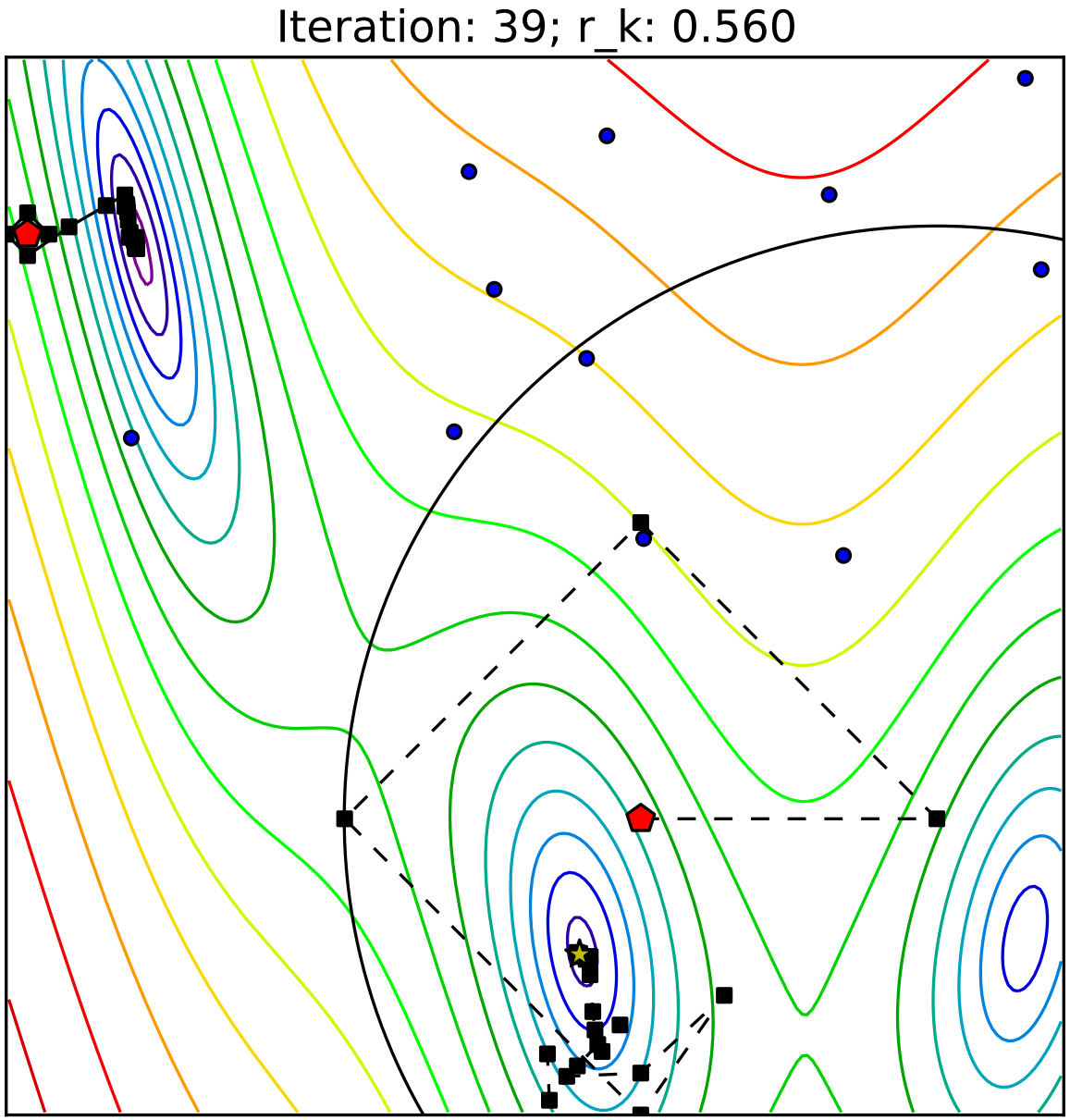
APOSMM



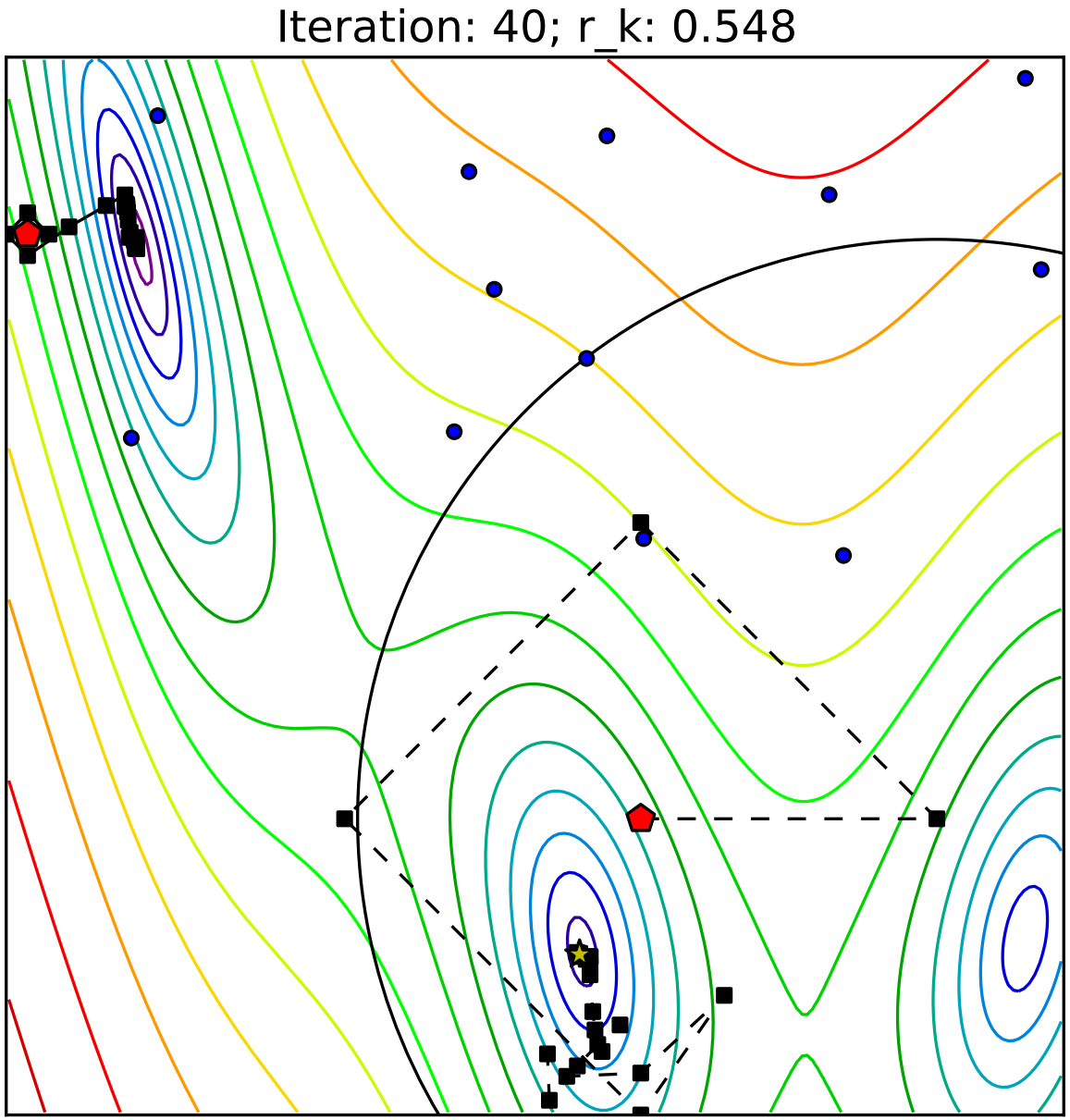
APOSMM



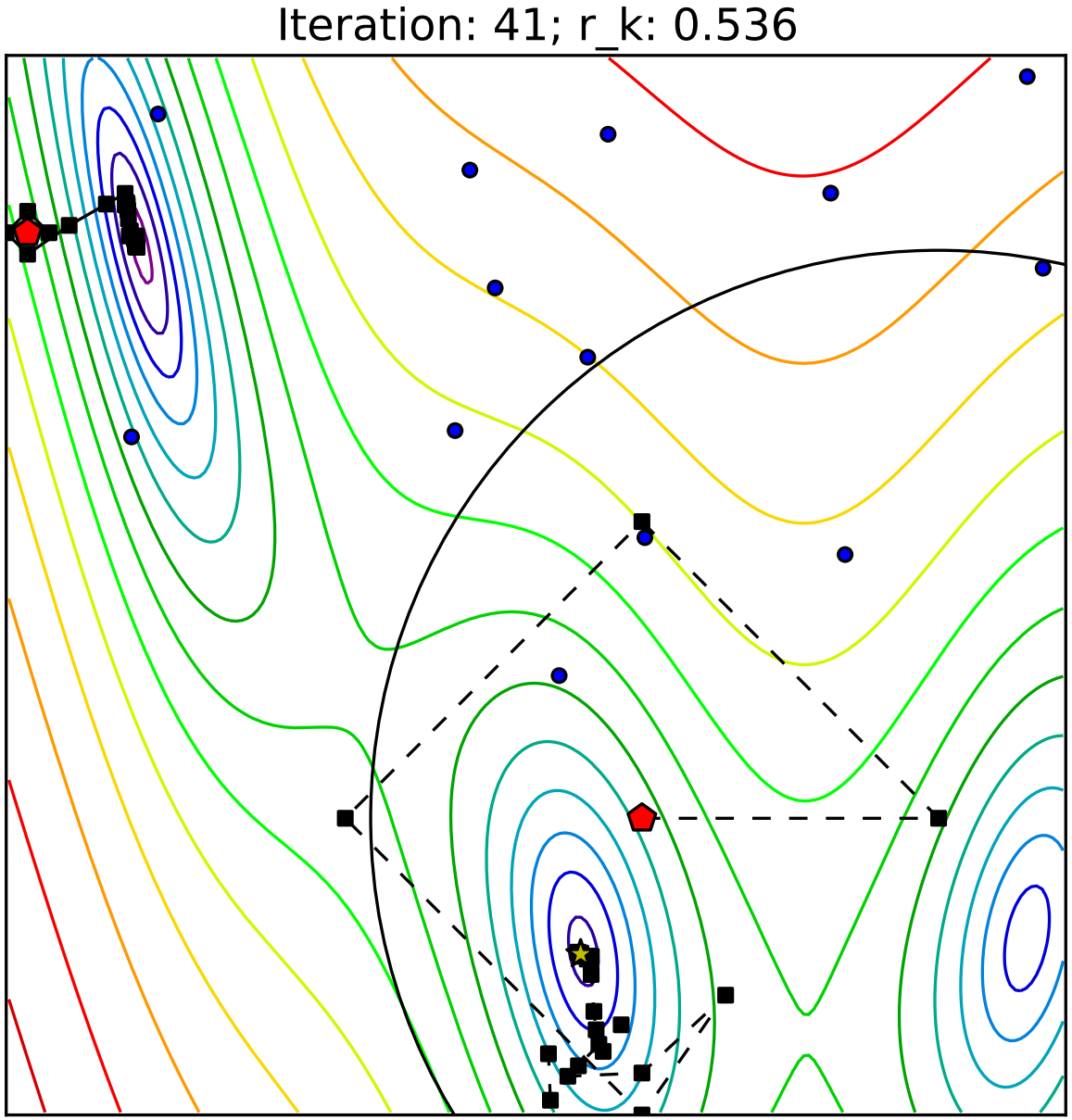
APOSMM



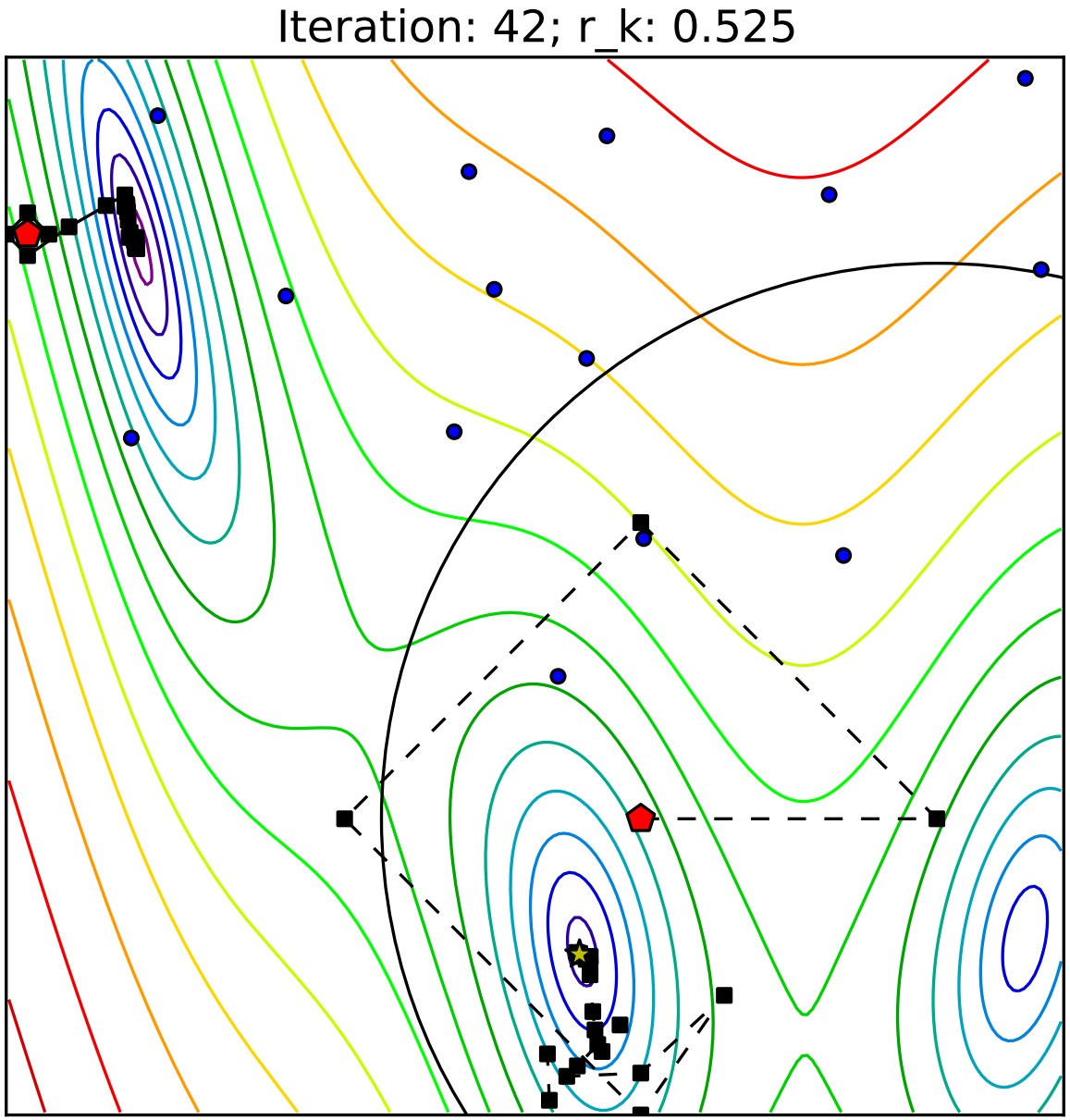
APOSMM



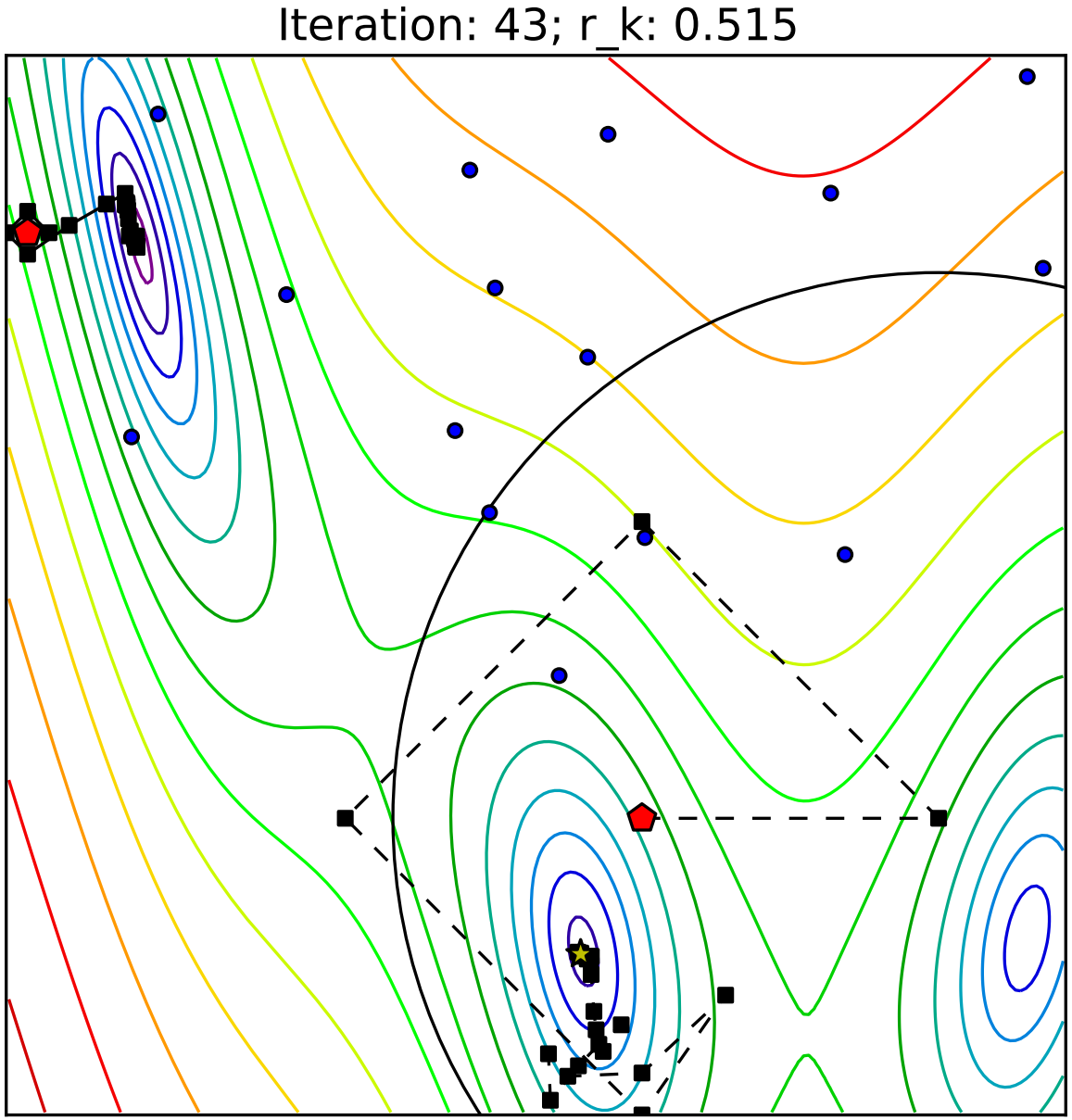
APOSMM



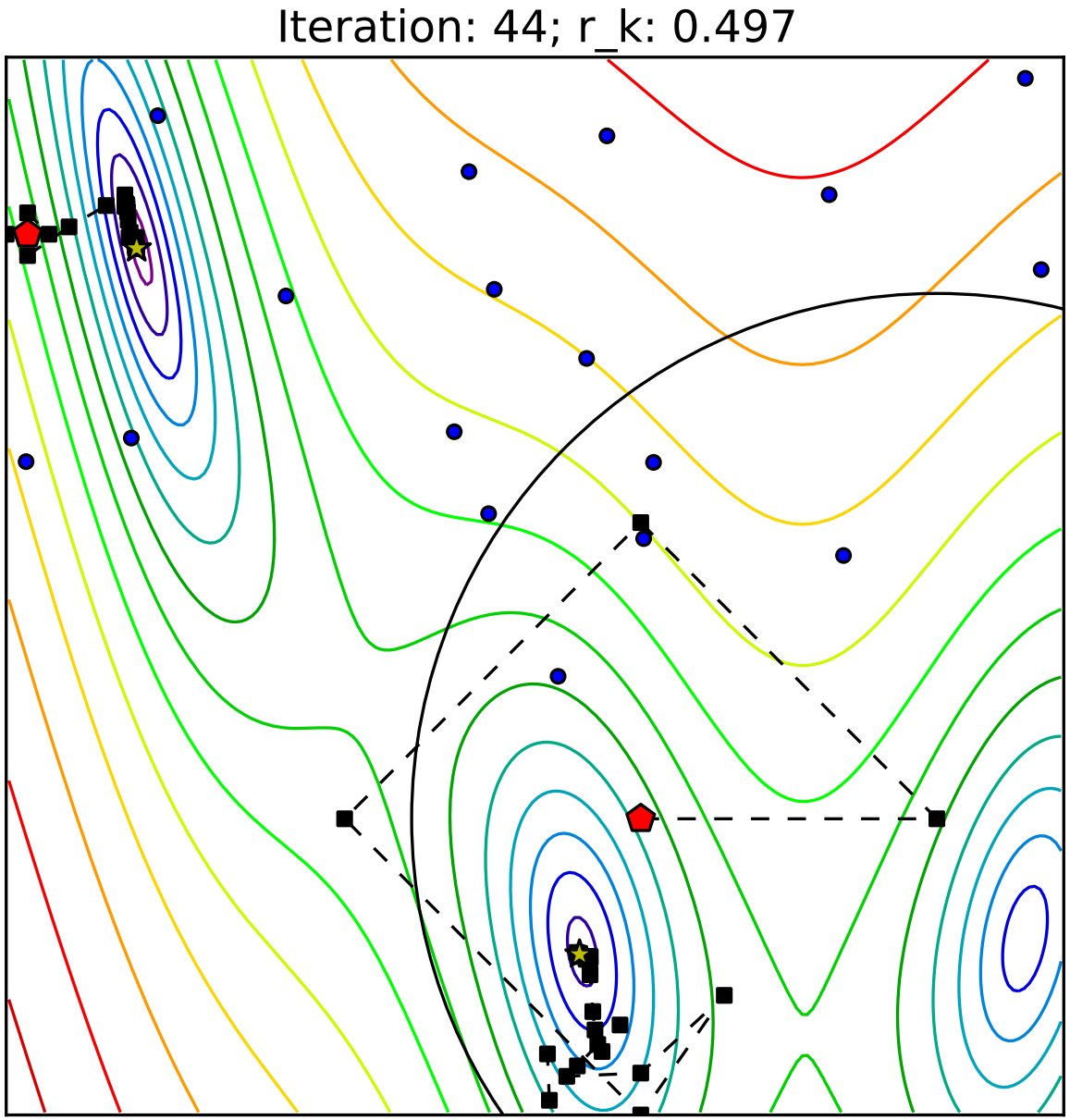
APOSMM



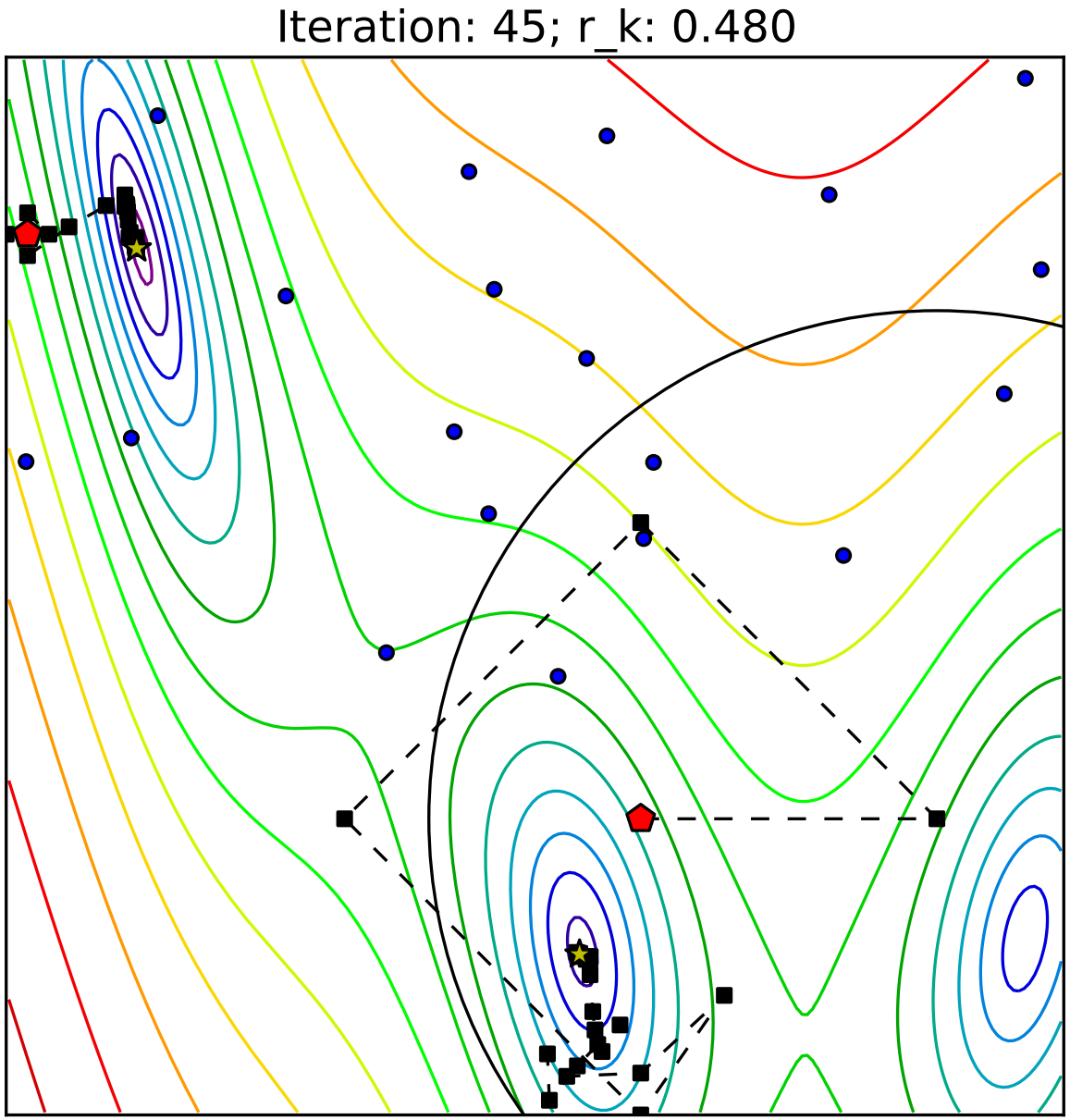
APOSMM



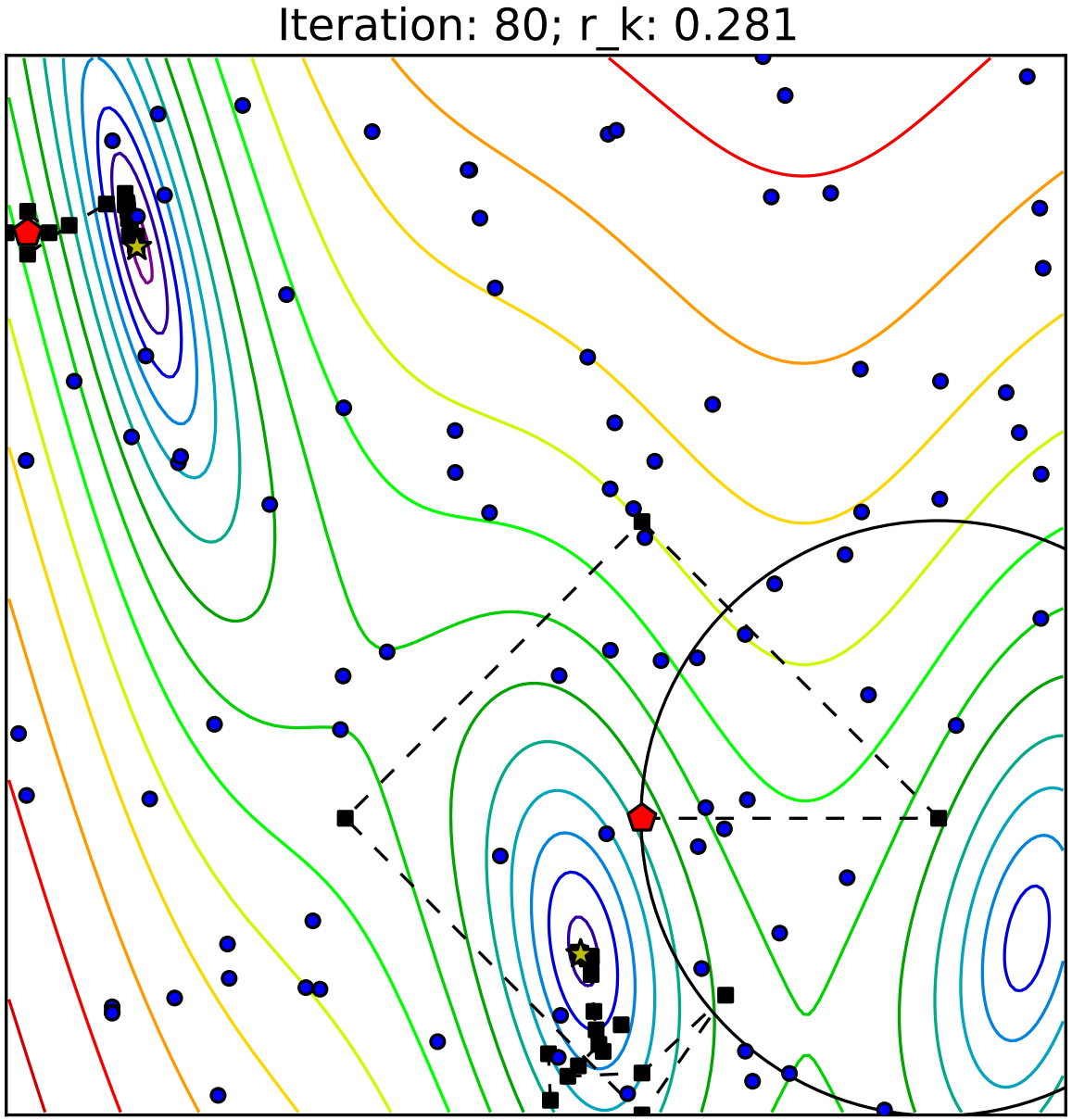
APOSMM



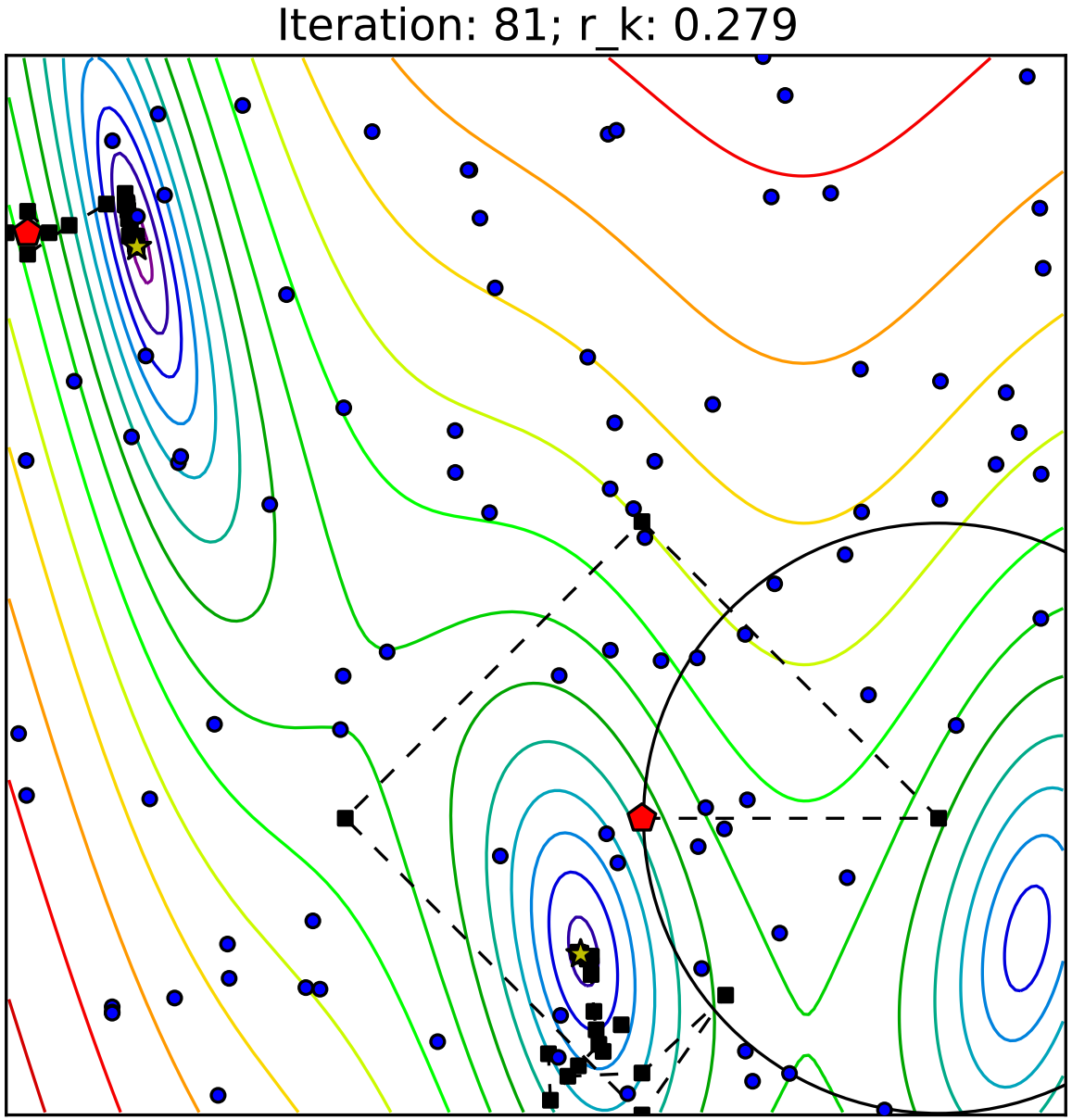
APOSMM



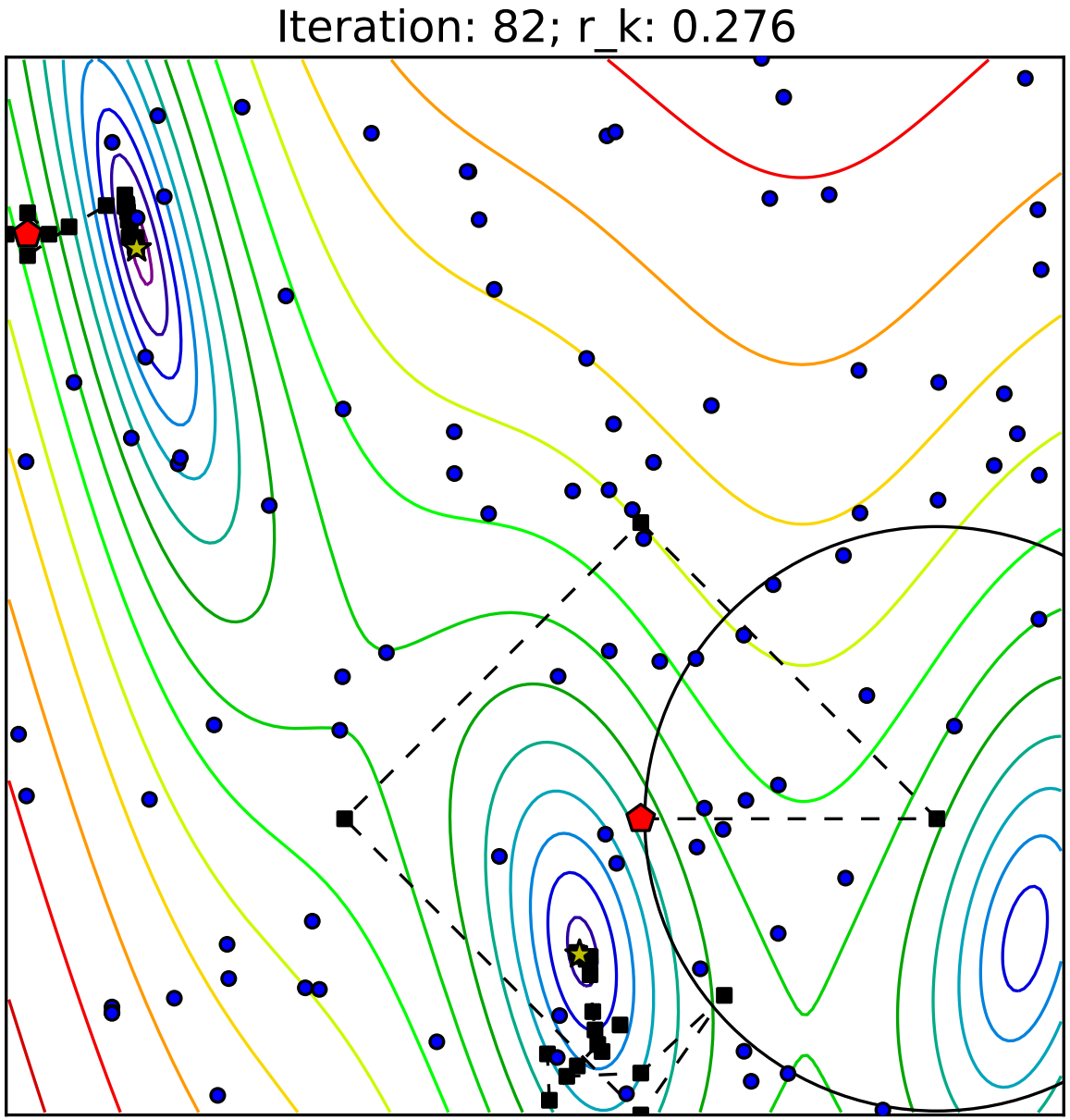
APOSMM



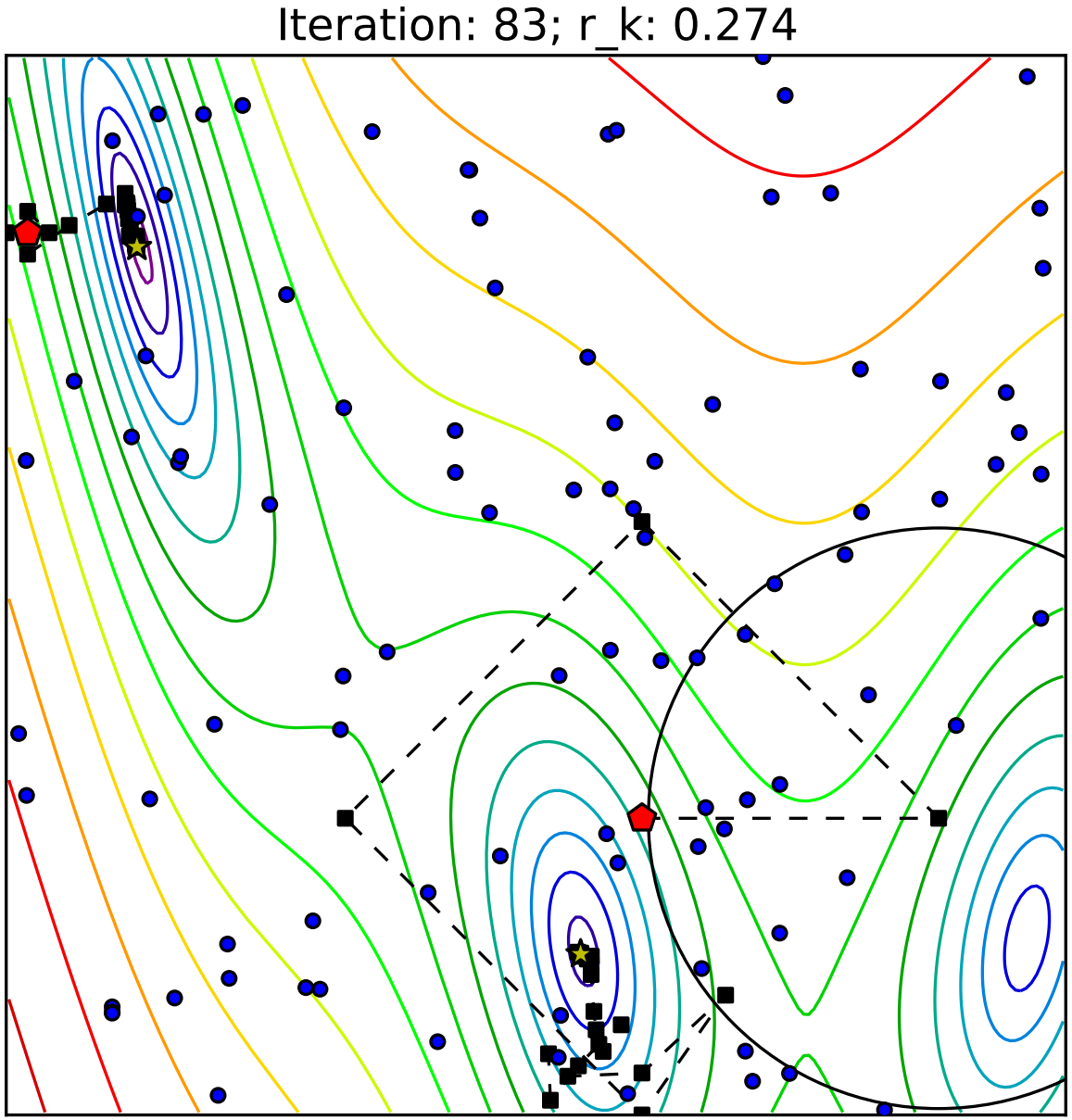
APOSMM



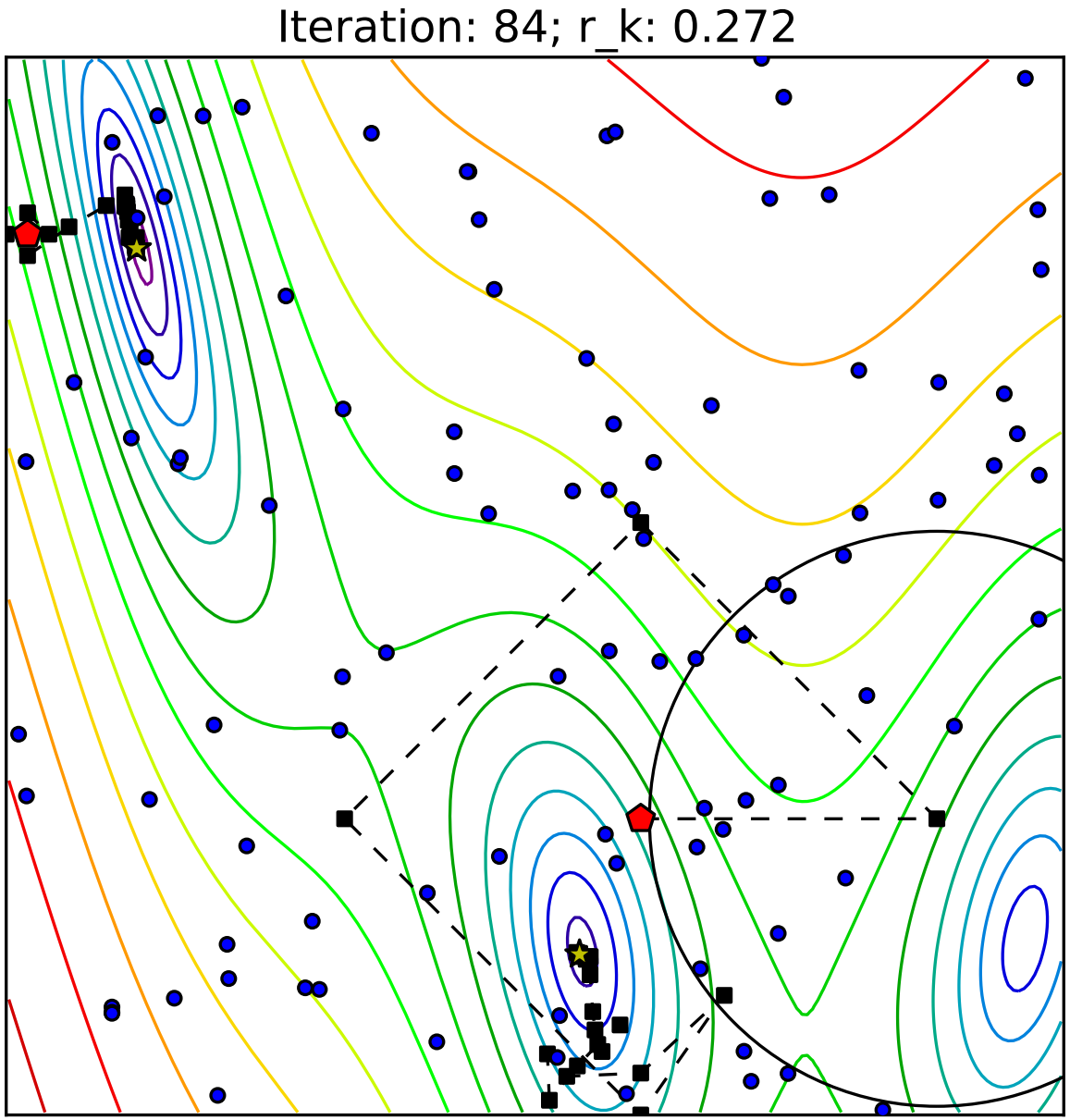
APOSMM



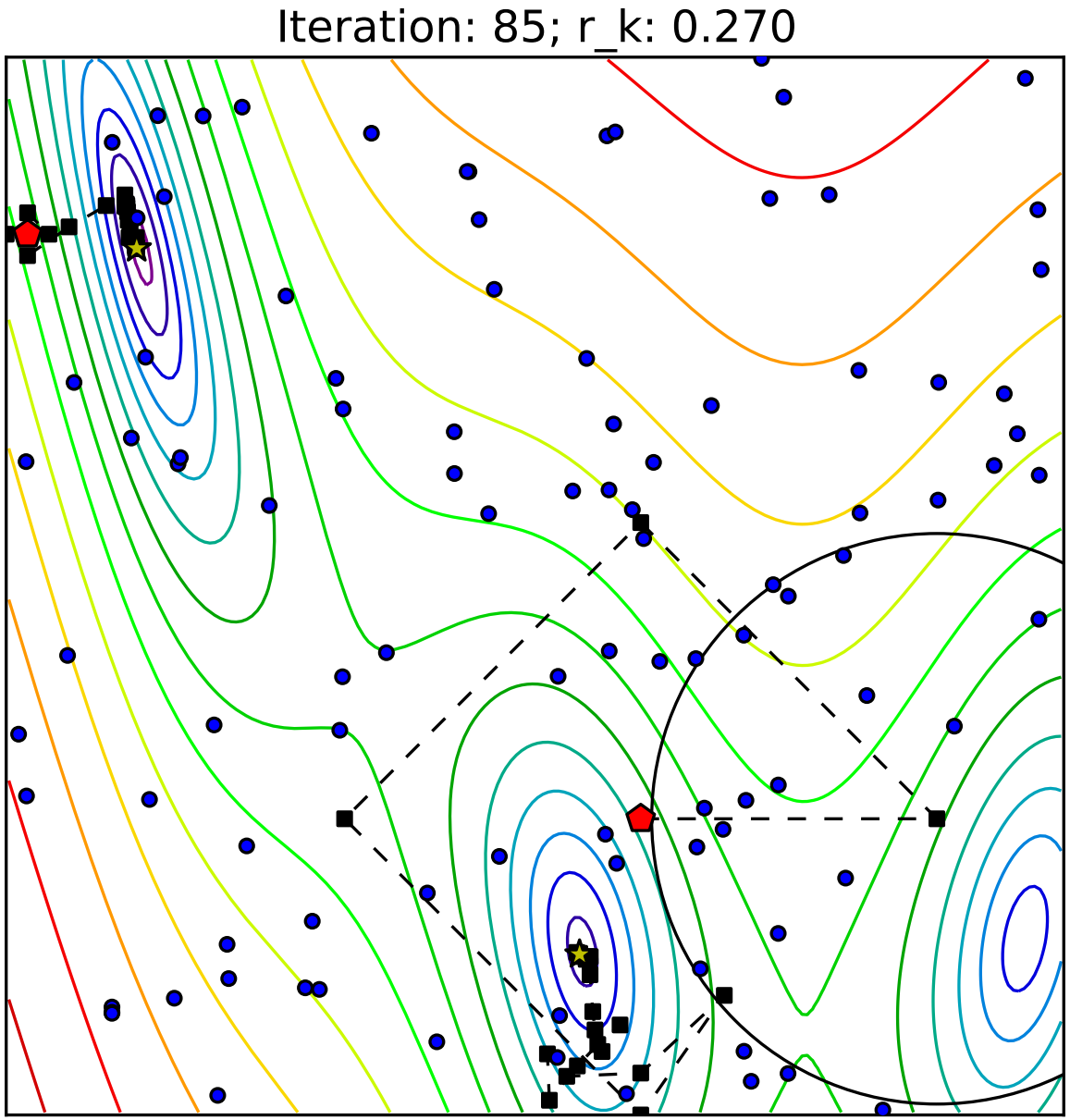
APOSMM



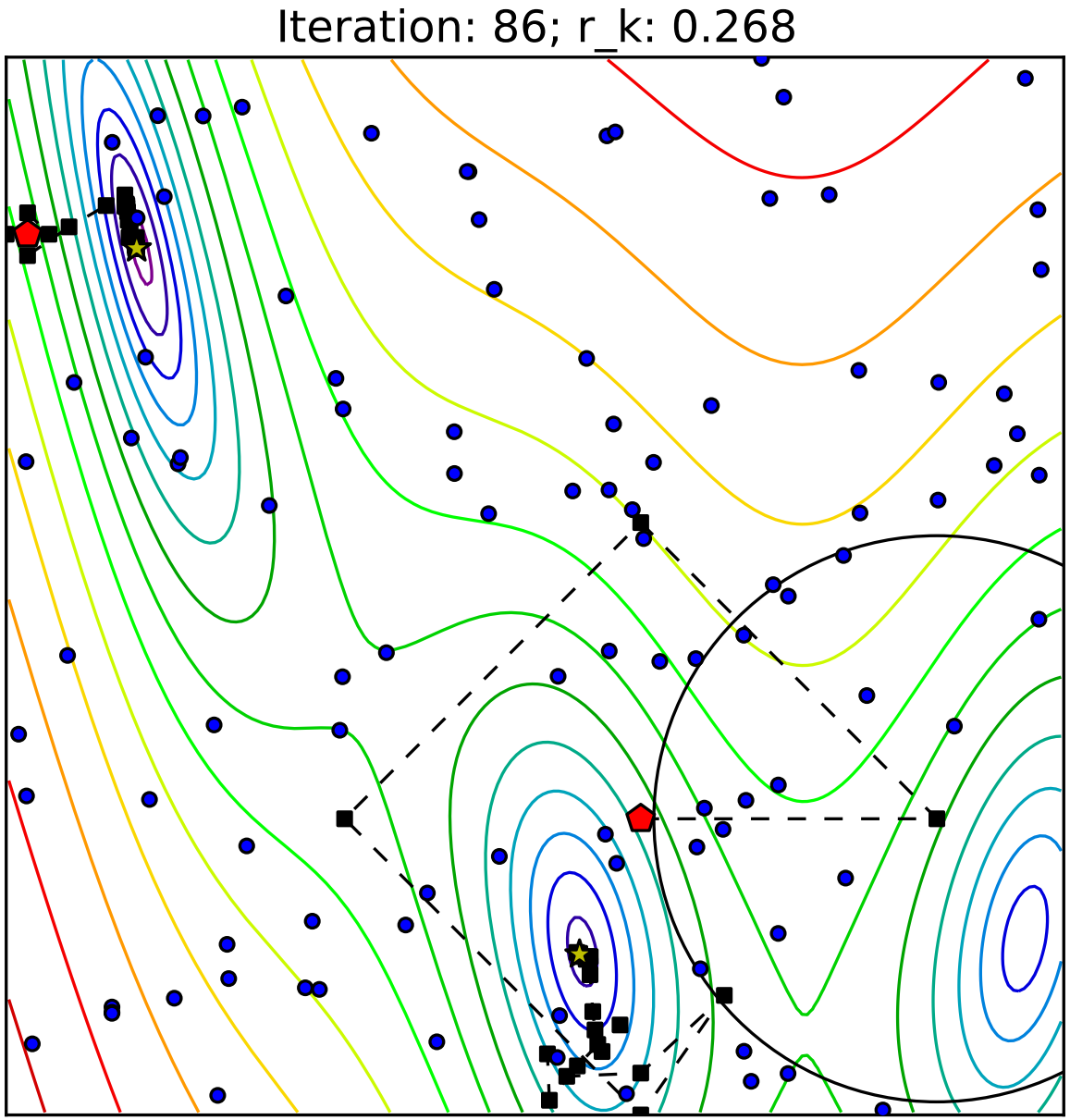
APOSMM



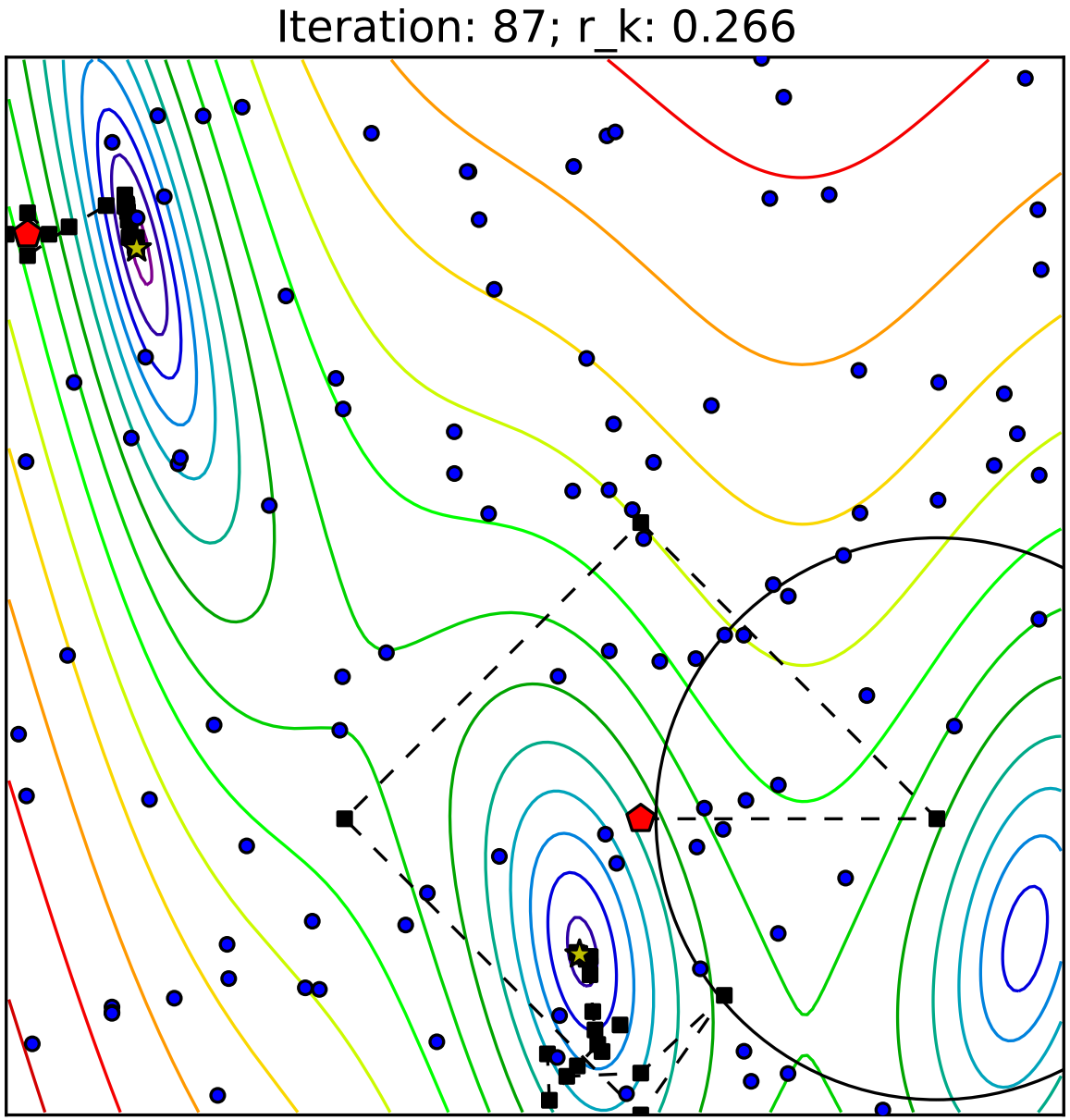
APOSMM



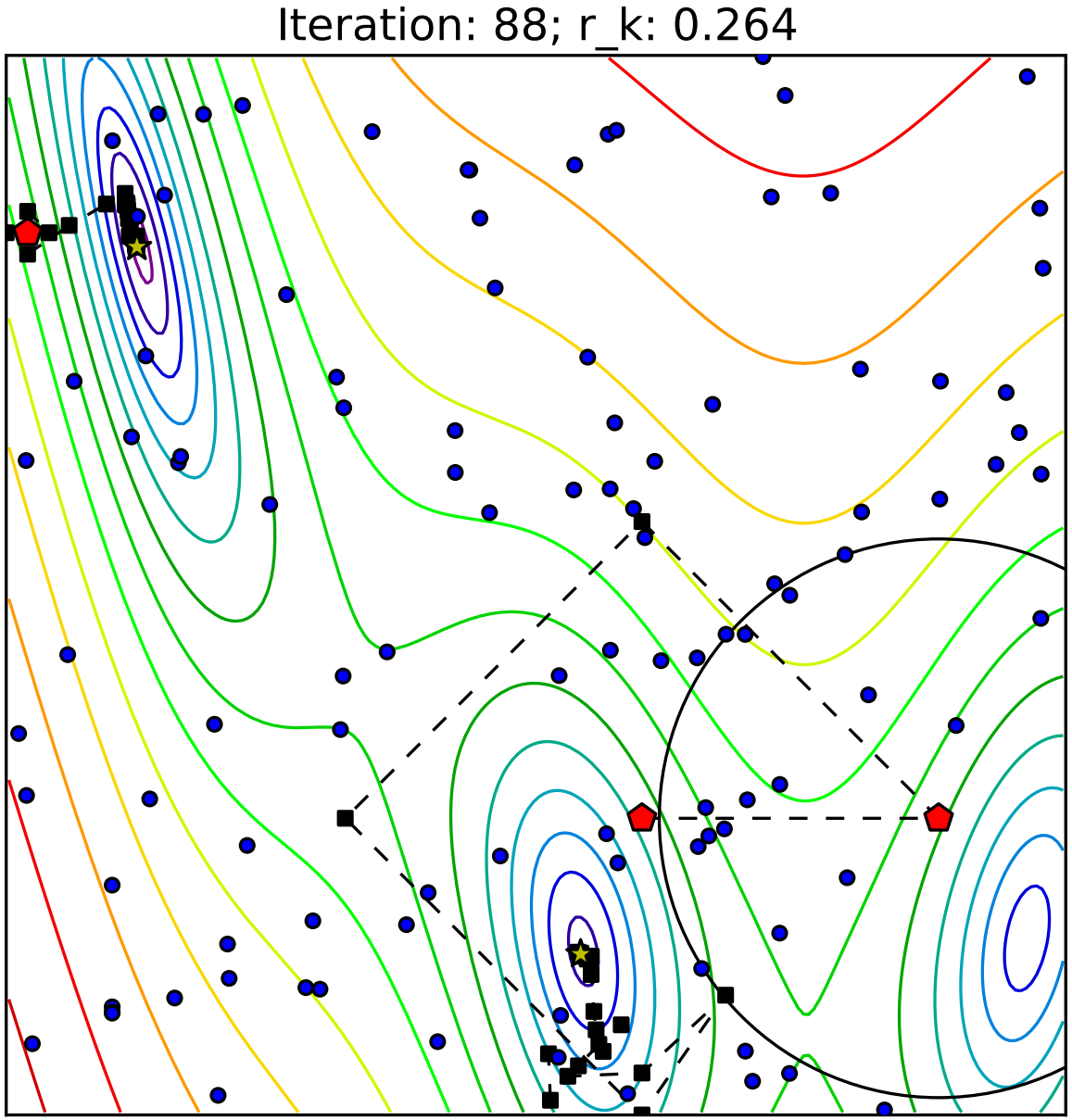
APOSMM



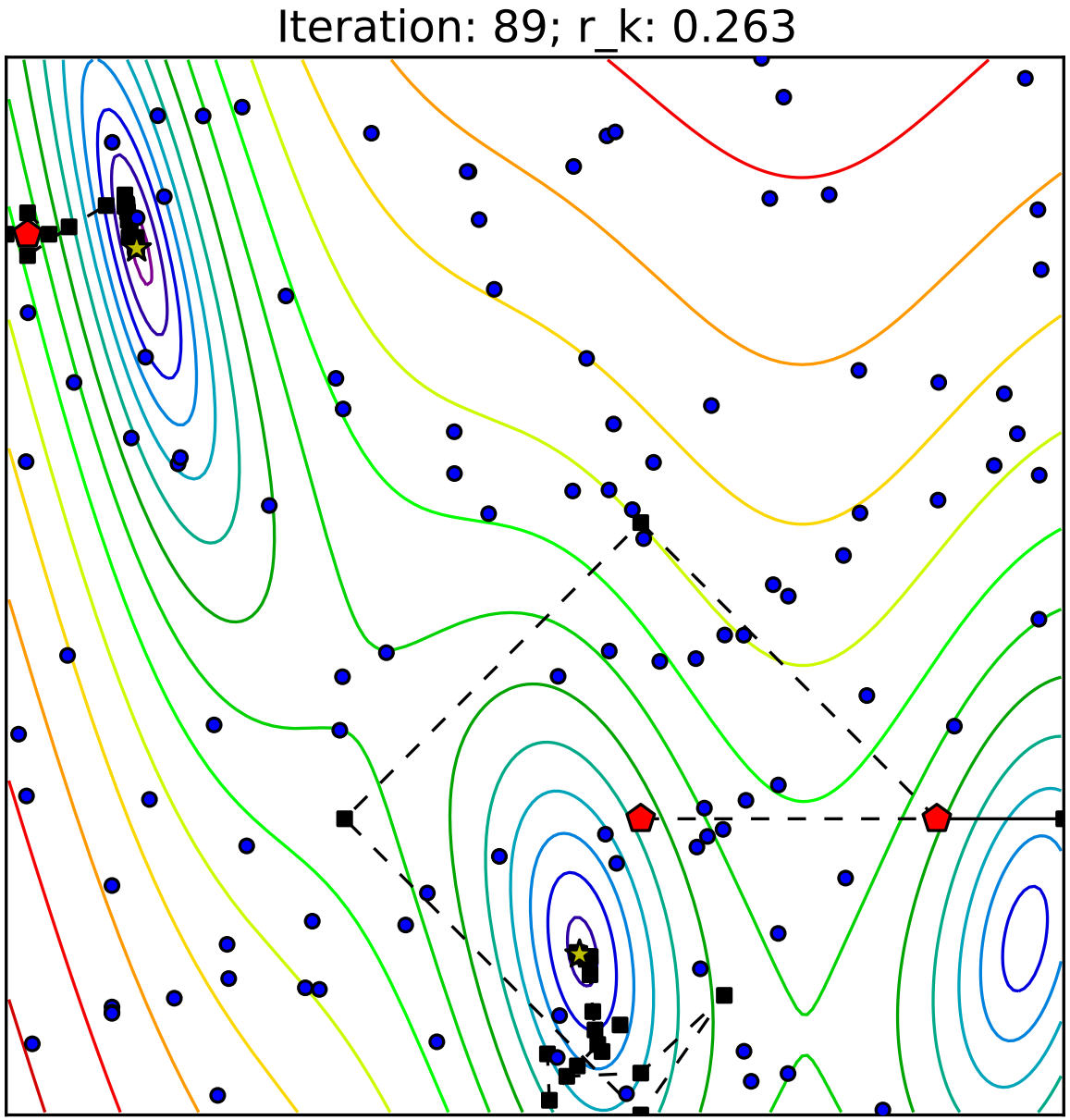
APOSMM



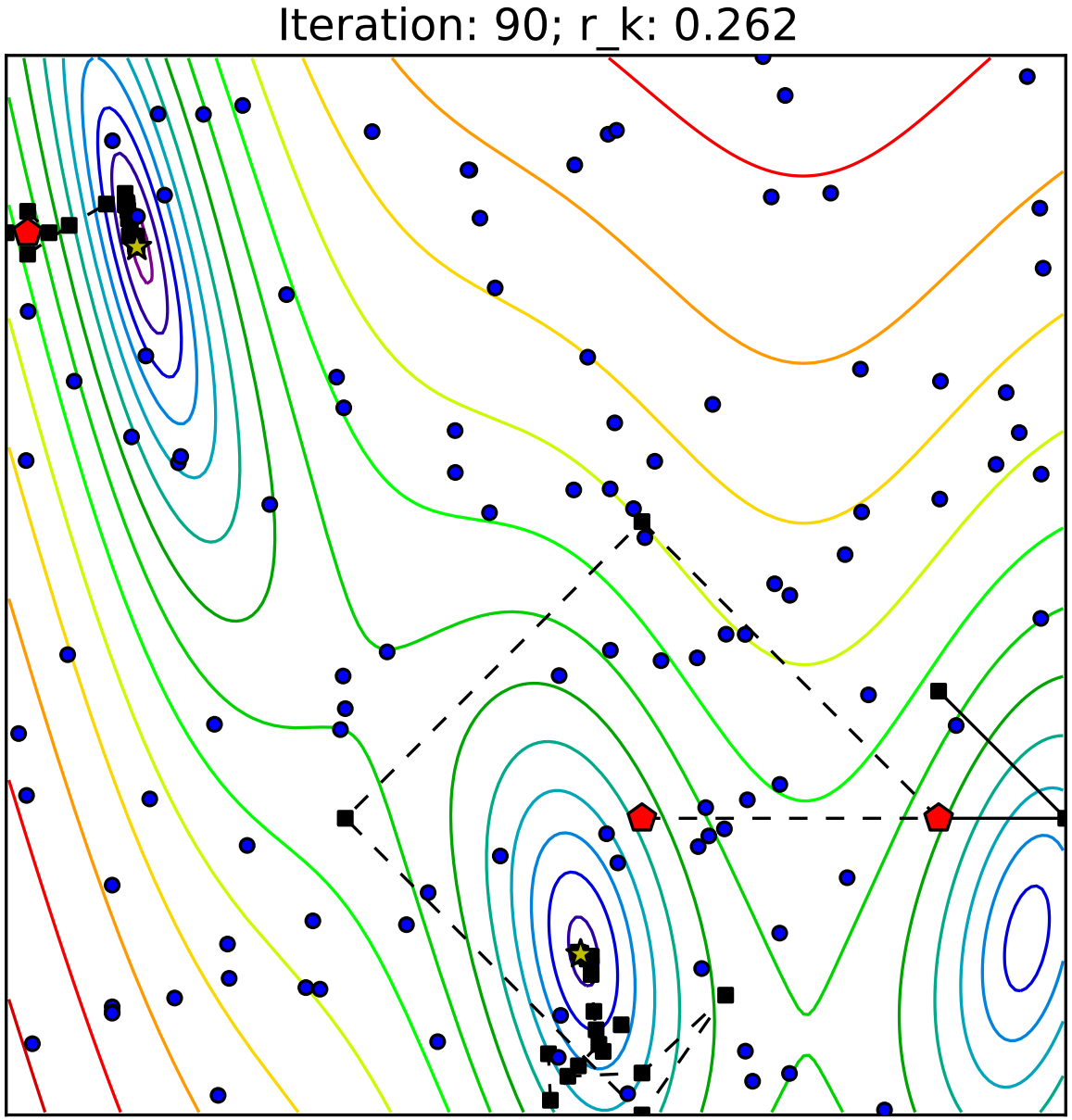
APOSMM



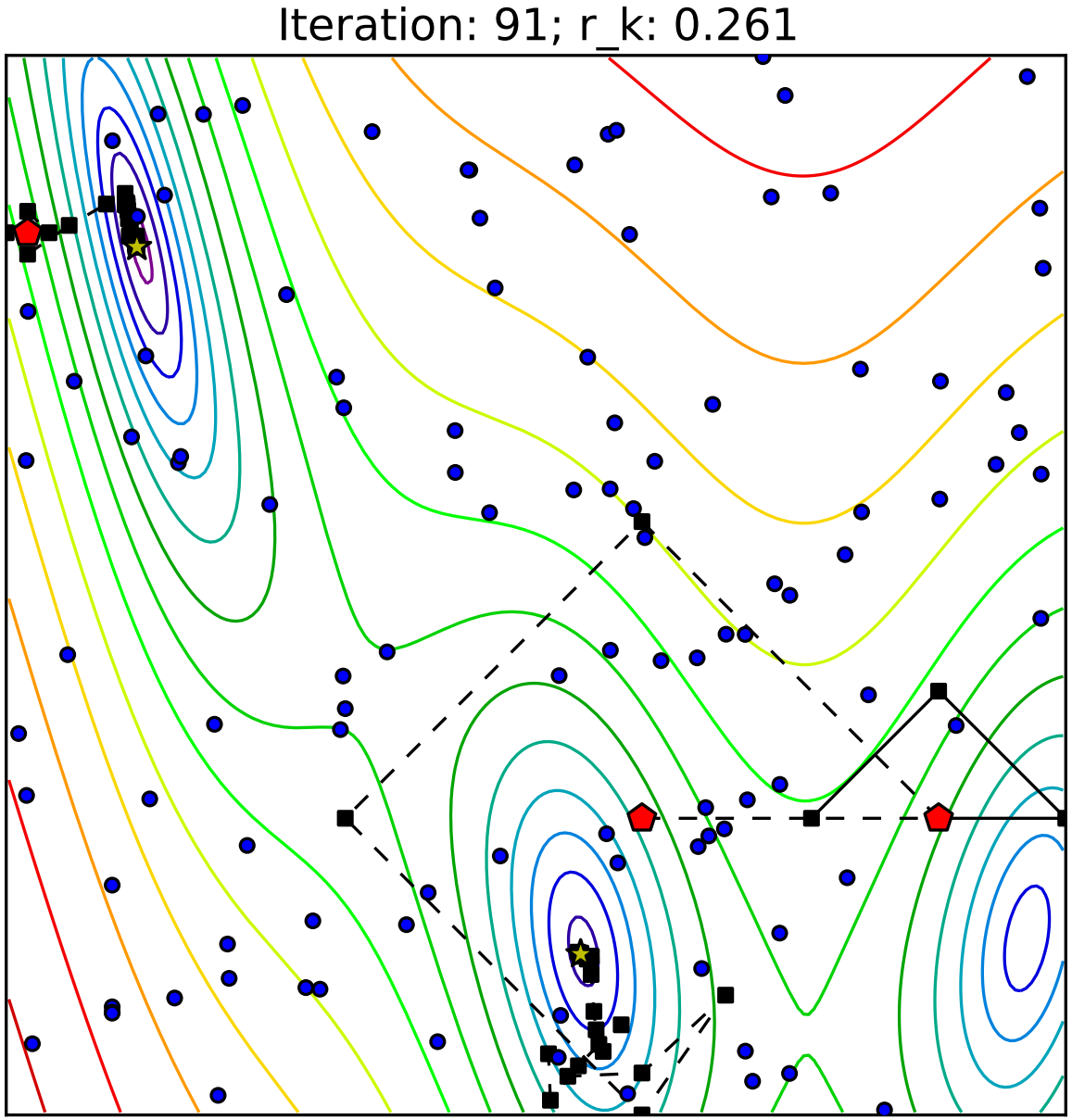
APOSMM



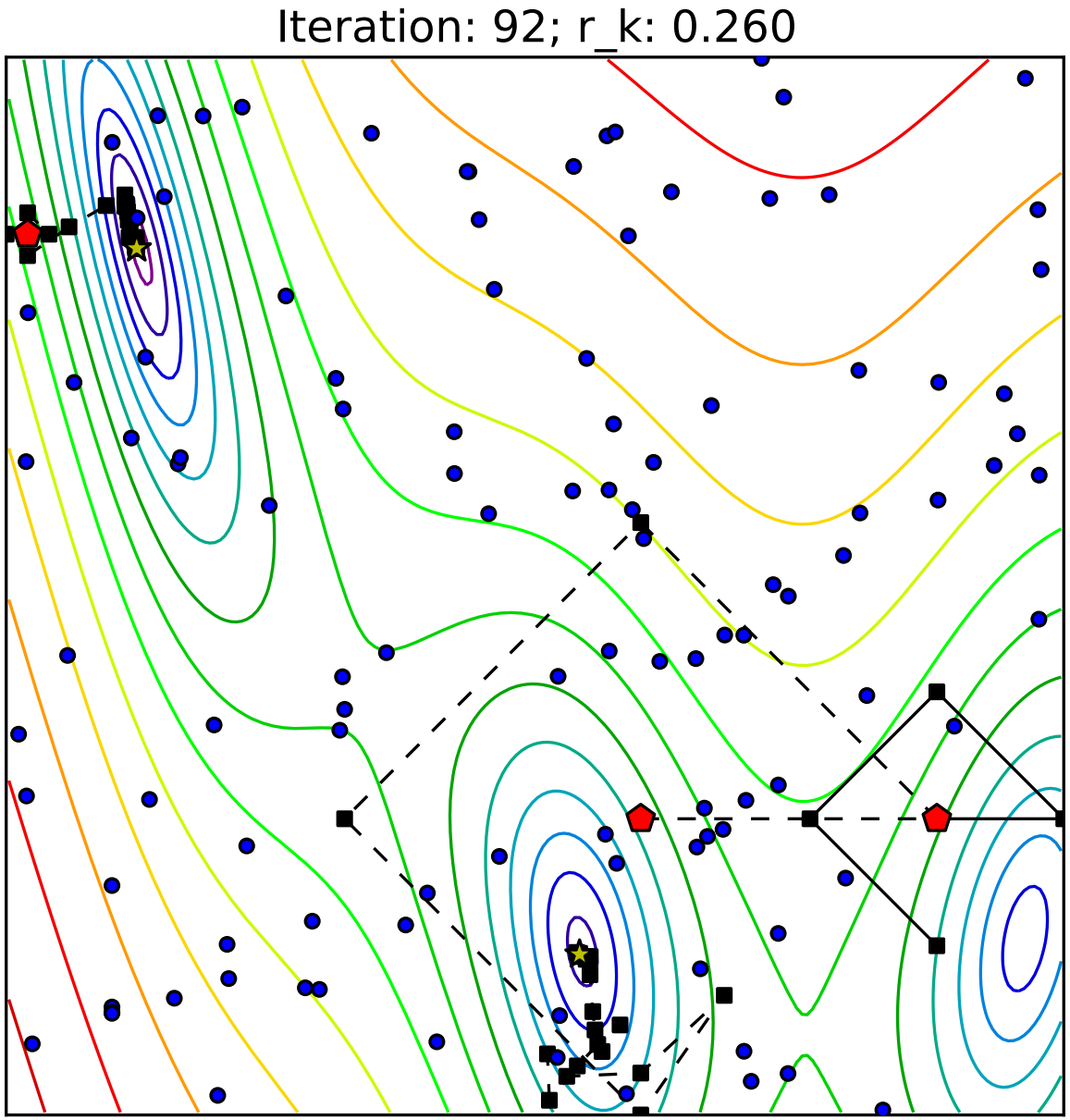
APOSMM



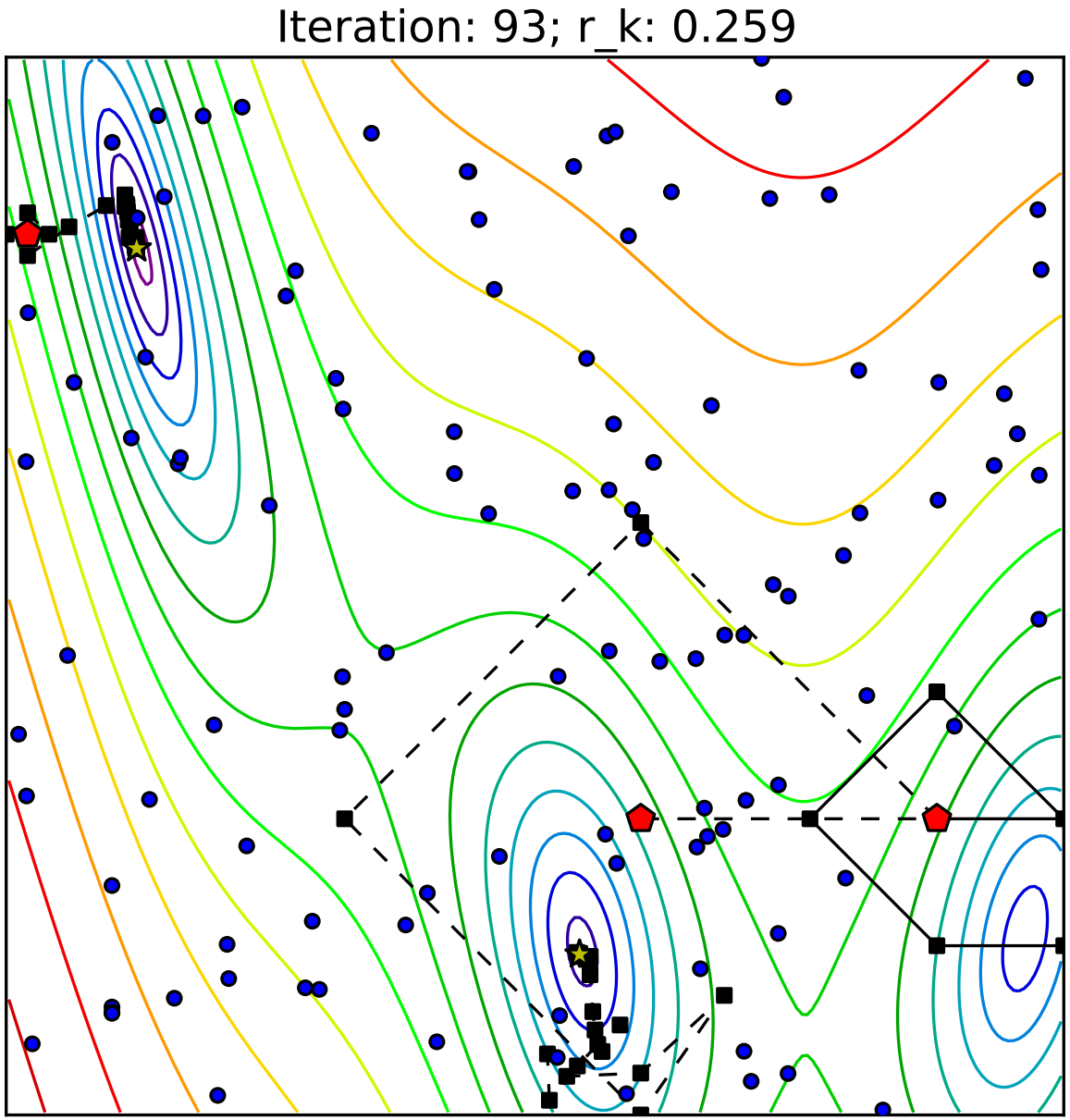
APOSMM



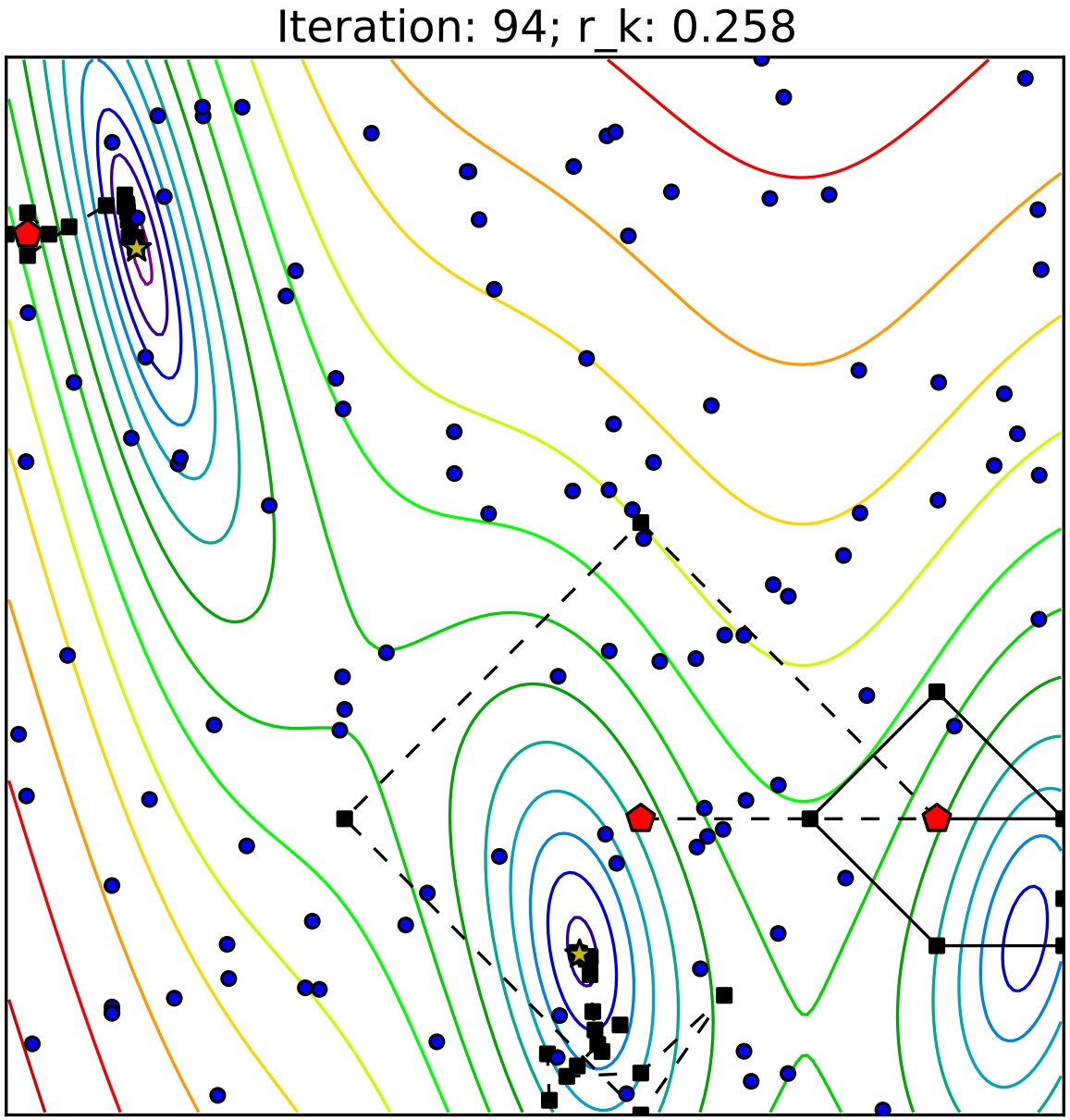
APOSMM



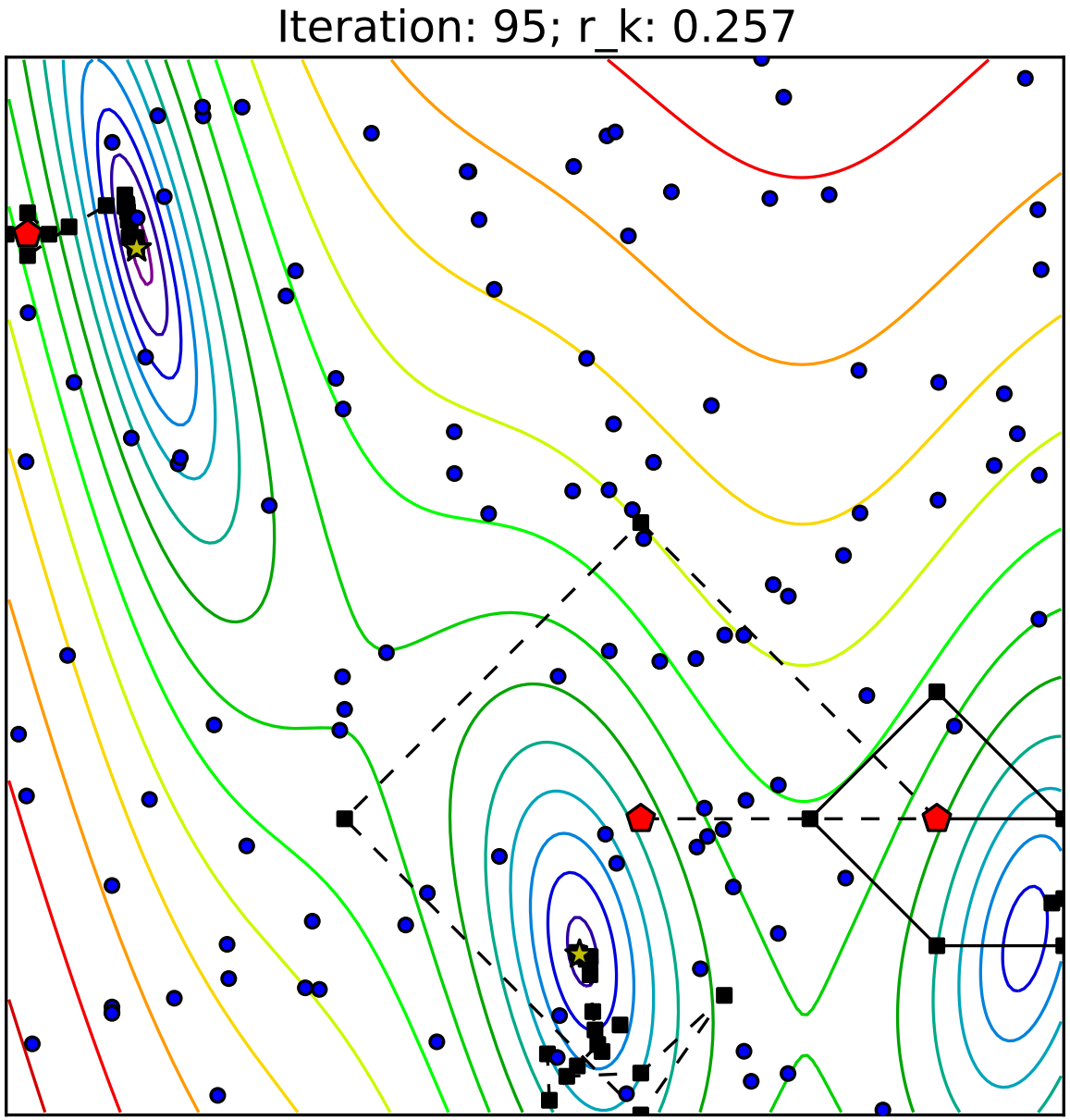
APOSMM



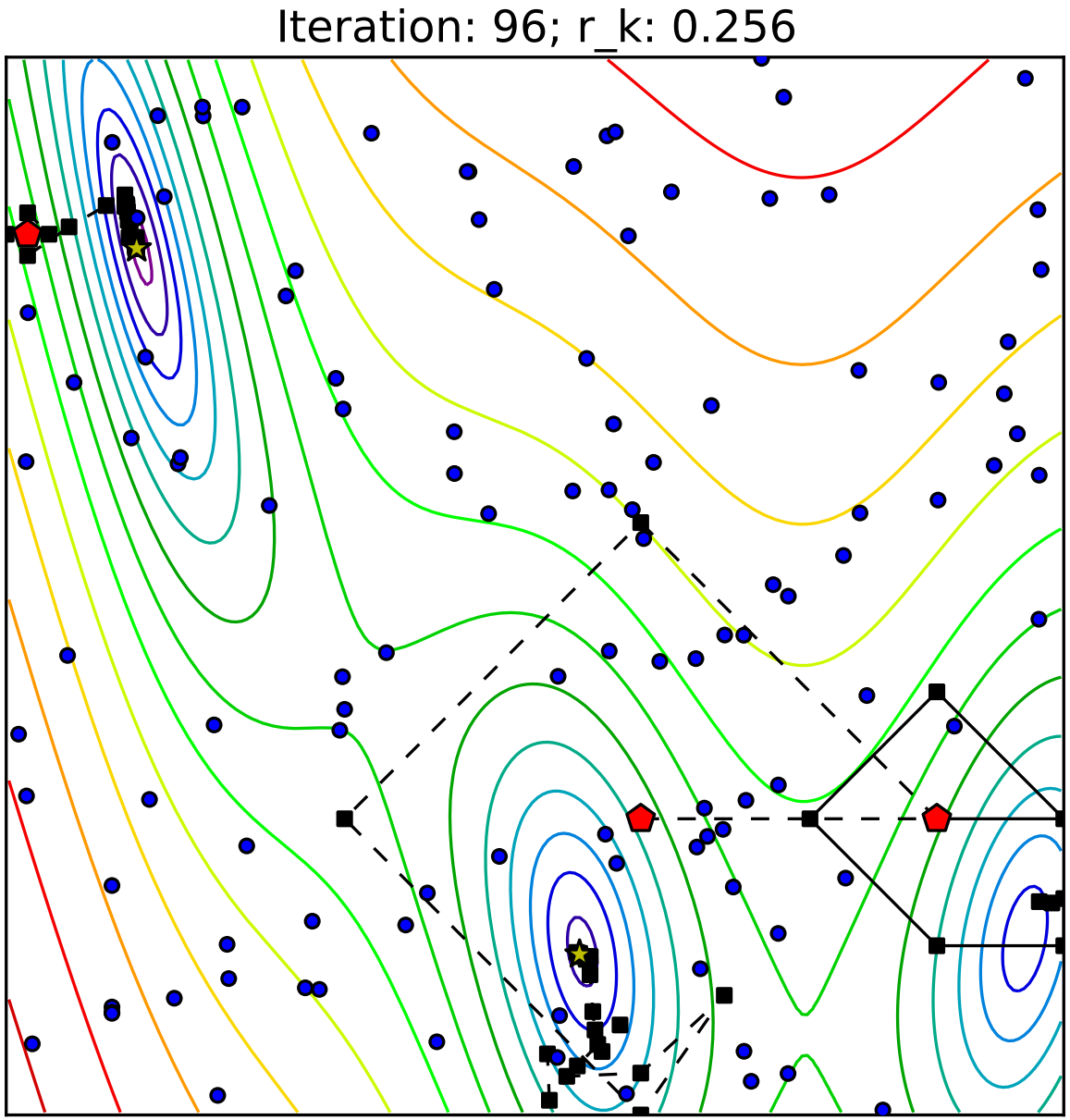
APOSMM



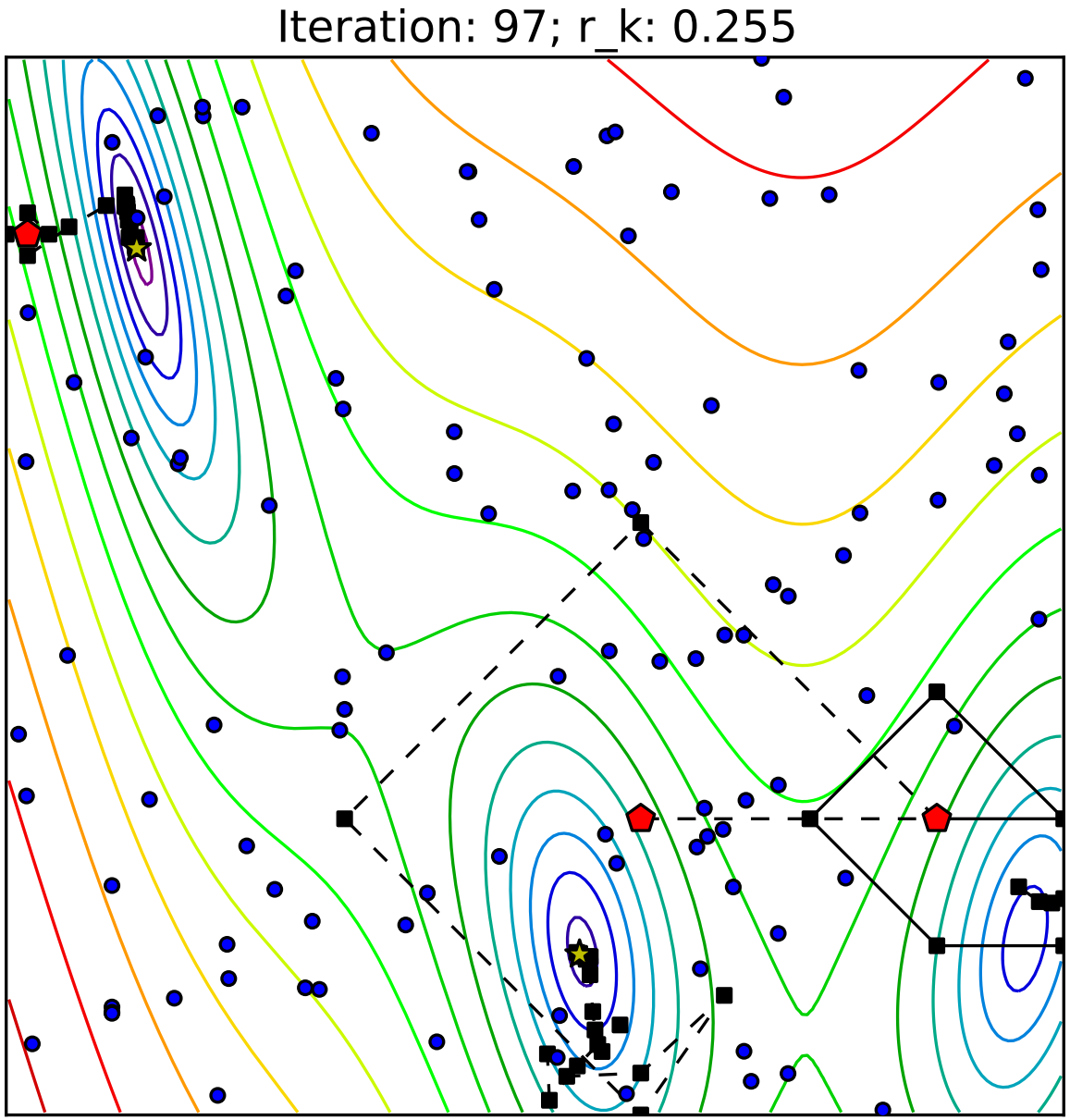
APOSMM



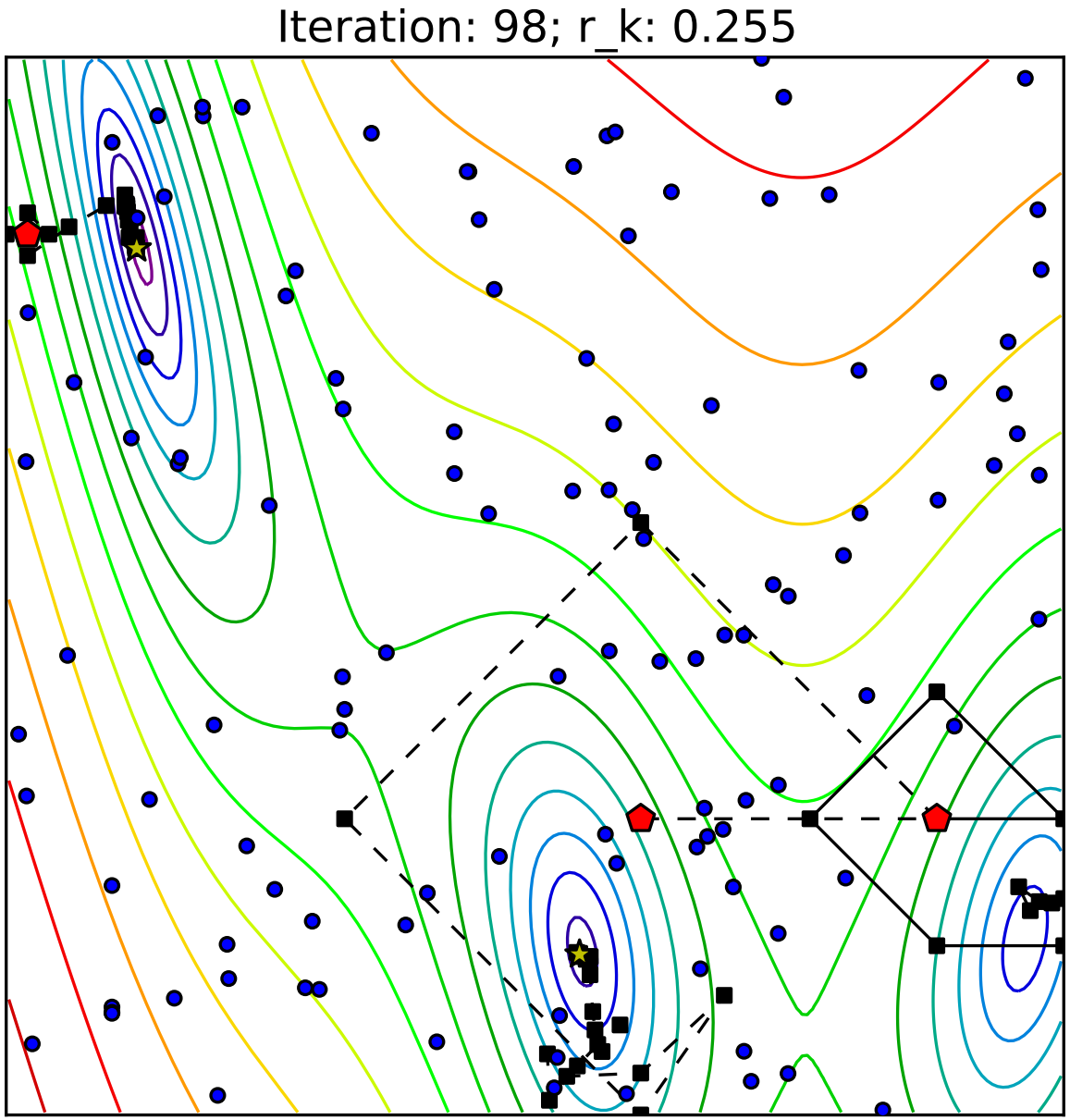
APOSMM



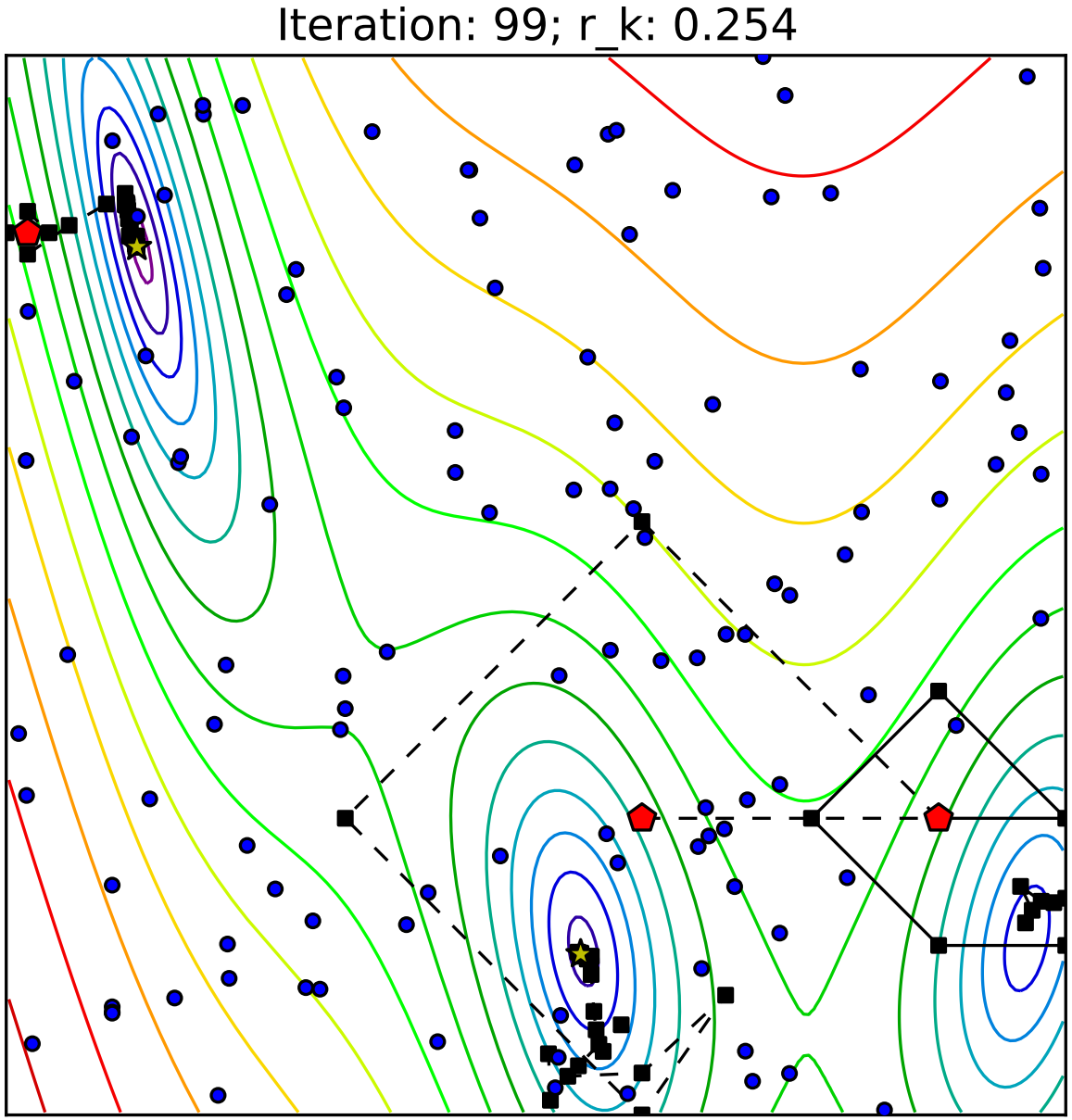
APOSMM



APOSMM



APOSMM



Properties of the local optimization method

Necessary:

- Honors a starting point
- Honors bound constraints

Properties of the local optimization method

Necessary:

- Honors a starting point
- Honors bound constraints

ORBIT satisfies these [Wild, Regis, Shoemaker, SIAM-JOSC, 2008]

BOBYQA satisfies these [Powell, 2009]

Properties of the local optimization method

Necessary:

- Honors a starting point
- Honors bound constraints

ORBIT satisfies these [Wild, Regis, Shoemaker, SIAM-JOSC, 2008]

BOBYQA satisfies these [Powell, 2009]

Possibly beneficial:

- Can return multiple points of interest
- Reports solution quality/confidence at every iteration
- Can avoid certain regions in the domain
- Uses a history of past evaluations of f
- Uses additional points mid-run

Exploiting Structure

- Nonsmooth, composite optimization

$$\underset{x}{\text{minimize}}\, f(x) = h(F(x))$$

  where ∇F is unavailable but ∂h is known

- Least-squares minimization

$$\underset{x}{\text{minimize}}\, f(x) = \sum_i (F_i(x) - T_i)^2$$

Theoretical results

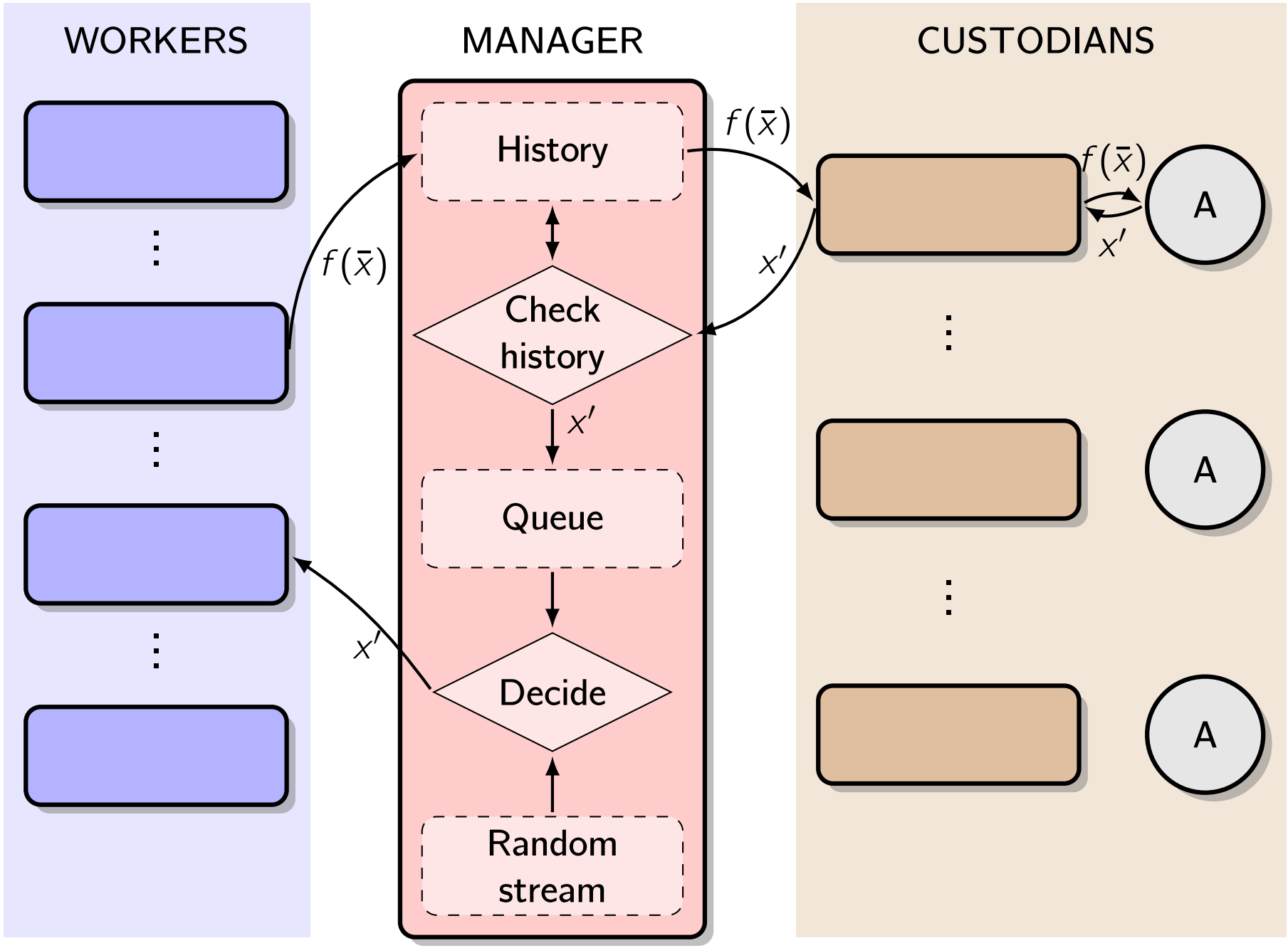
- $f \in C^2$, with local minima in the interior of $\mathcal{D}$, and the distance between minima is bounded away from zero.
- $\mathcal{L}$ is strictly descent and converges to a minimum (not a stationary point).
- Each run has points evaluated often enough
- $$r_k = \frac{1}{\sqrt{\pi}} \sqrt[n]{\Gamma\left(1 + \frac{n}{2}\right) \operatorname{vol}(\mathcal{D}) \frac{5 \log |\mathcal{S}_k|}{|\mathcal{S}_k|}}$$

Theoretical results

- $f \in C^2$, with local minima in the interior of $\mathcal{D}$, and the distance between minima is bounded away from zero.
- $\mathcal{L}$ is strictly descent and converges to a minimum (not a stationary point).
- Each run has points evaluated often enough
- $$r_k = \frac{1}{\sqrt{\pi}} \sqrt[n]{\Gamma\left(1 + \frac{n}{2}\right) \operatorname{vol}(\mathcal{D}) \frac{5 \log |\mathcal{S}_k|}{|\mathcal{S}_k|}}$$

Theorem

Each minima will almost surely be identified: either found in a finite number of evaluations or have a single local optimization run converging asymptotically to it.

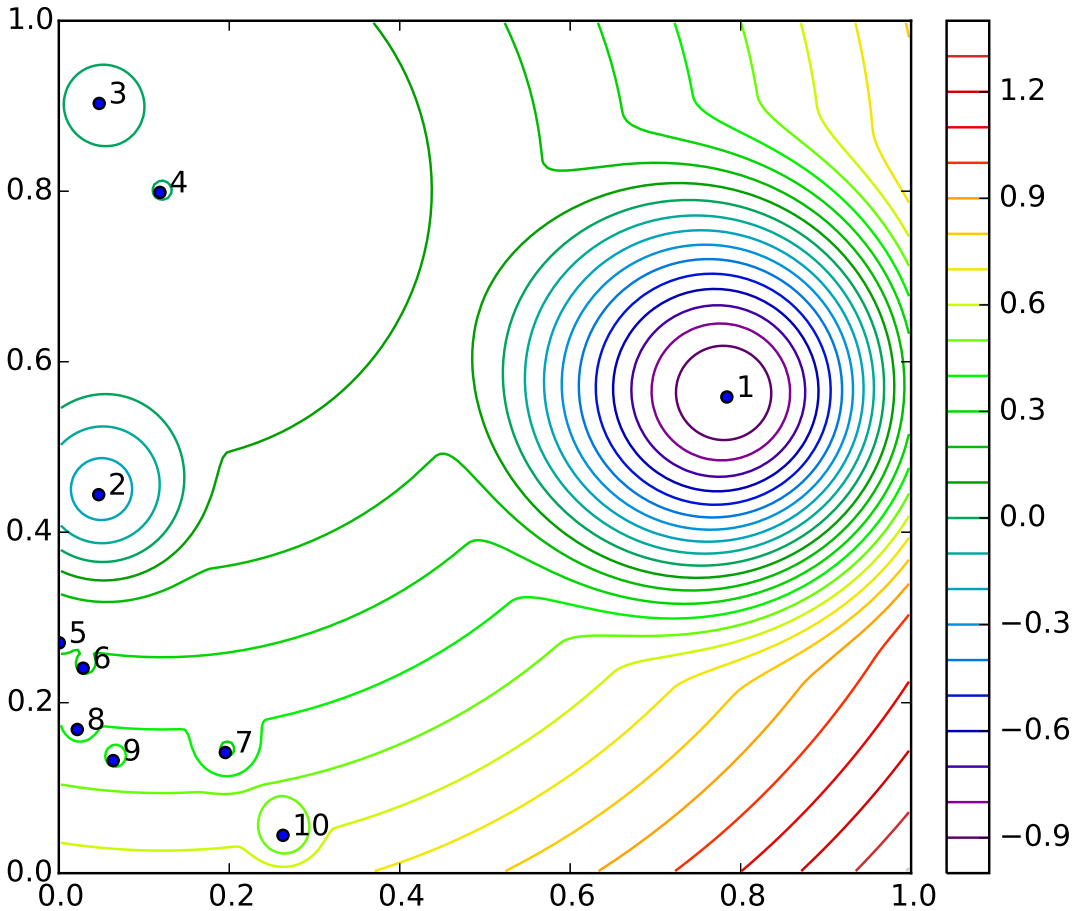

WORKERS
MANAGER
CUSTODIANS
History
Check history
Queue
Decide
Random stream
$f(\bar{x})$
x'
A

Measuring Performance

GLODS	Global & local optimization using direct search [Custódio, Madeira (JOGO, 2014)]
Direct	Serial Matlab DiRect code [Finkel (2003)]
pVTdirect	Parallel DiRect code [He, Watson, Sosonkina (TOMS, 2009)]
CMA-ES	Parallel Covariance Matrix Adaptation Evolution Strategy code [Hansen & Ostermeier (EvolComp, 2001)]
Random	Uniform sampling (as a baseline)
(A)POSMM	[Larson, W. (2016)] with local solver BOBYQA [Powell (2009)], Initial sample size: $10n$

GKLS problems [Gaviano et al., "Algorithm 829" (TOMS, 2003)]

- Smooth augmentation of a convex quadratic; $n = 2, \ldots, 7$
- 10 local minima in the unit cube with a unique global minimum
- 100 problems for each n; 5 replications; $2000(n+1)$ evaluations



Measuring Performance

Notation:

Let X^* be the set of all local minima of f.
Let $f^*_{(i)}$ be the ith smallest value $\{f(x^*)|x^* \in X^*\}$.
Let $x^*_{(i)}$ be the element of X^* corresponding to the value $f^*_{(i)}$.

The global minimum has been found at a level $\tau > 0$ after k evaluations if an algorithm it has found a point $\hat{x}$ satisfying:

$$f(\hat{x}) - f^*_{(1)} \leq (1 - \tau)\left(f(x_0) - f^*_{(1)}\right),$$

where x_0 is the starting point for problem p.

Measuring Performance

Notation:

Let X^* be the set of all local minima of f.
Let $f^*_{(i)}$ be the ith smallest value $\{f(x^*)|x^* \in X^*\}$.
Let $x^*_{(i)}$ be the element of X^* corresponding to the value $f^*_{(i)}$.

The j best local minima have been found at a level $\tau > 0$ after k evaluations if:

$$\left|\left\{x^*_{(1)}, \ldots, x^*_{(\underline{j}-1)}\right\} \bigcap \left\{x^*_{(i)} : \exists x \in \mathcal{H}_k \text{ with } \left\|x - x^*_{(i)}\right\| \leq r_n(\tau)\right\}\right| = \underline{j} - 1$$

&

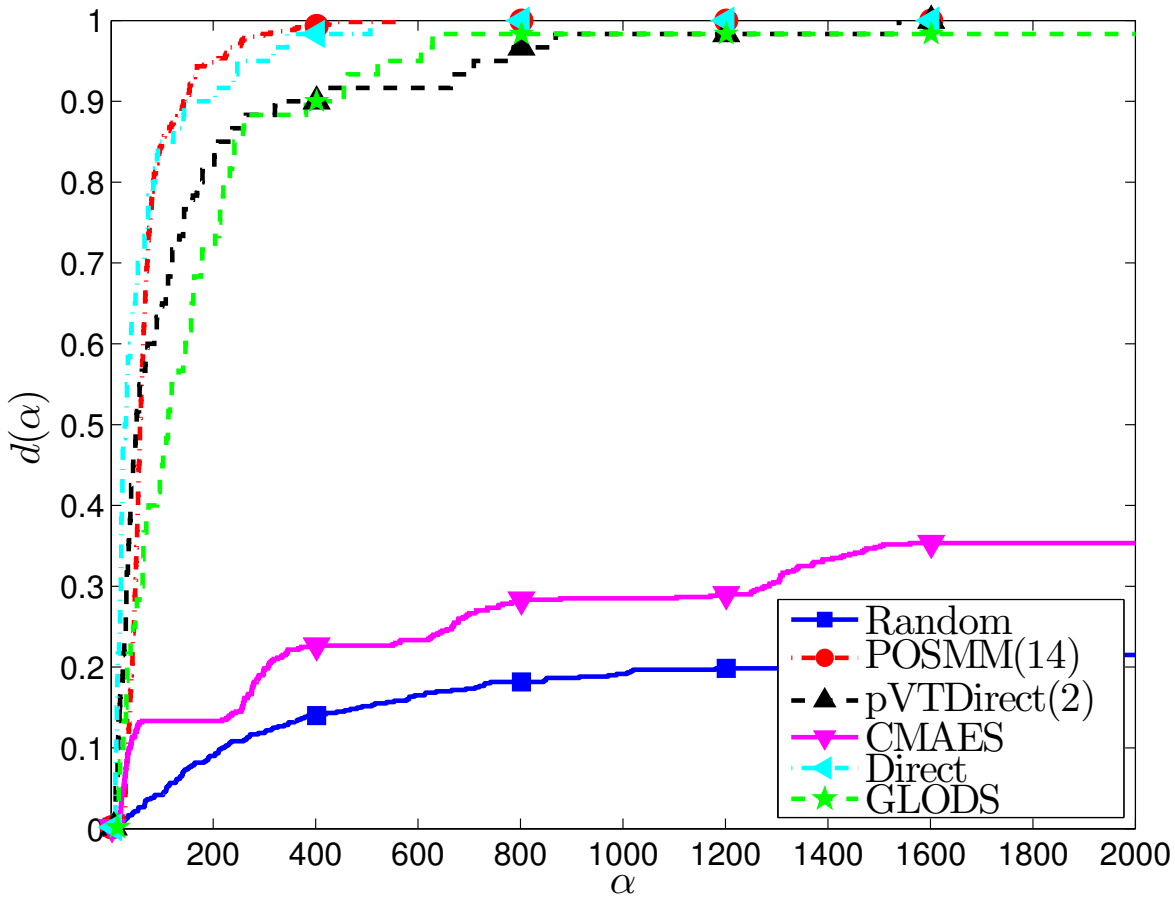
$$\left|\left\{x^*_{(\underline{j})}, \ldots, x^*_{(\bar{j})}\right\} \bigcap \left\{x^*_{(i)} : \exists x \in \mathcal{H}_k \text{ with } \left\|x - x^*_{(i)}\right\| \leq r_n(\tau)\right\}\right| \geq j - \underline{j} + 1,$$

where $\underline{j}$ and $\bar{j}$ are the smallest and largest integers such that $f^*_{(\underline{j})} = f^*_{(j)} = f^*_{(\bar{j})}$ and where $r_n(\tau) = \sqrt[n]{\frac{\tau \, \text{vol}(\mathcal{D})\Gamma(\frac{n}{2}+1)}{\pi^{n/2}}}$.

Ability to Find Approximate Global Minimizer

(A)POSMM

- Makes rapid progress to f_G
- Outperforms other algorithms *(even while demanding 14-fold concurrency)*



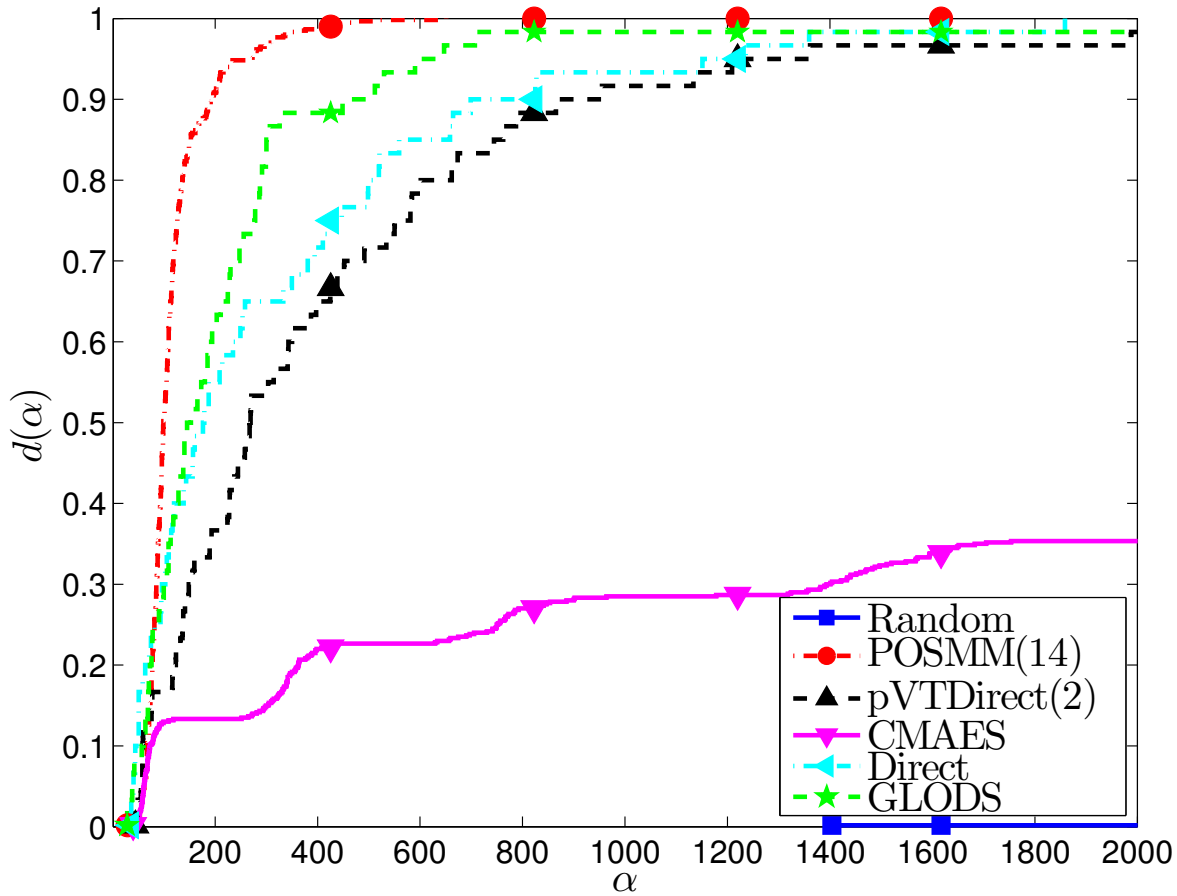
$$\tau = 10^{-2}$$

$$f(x) - f_G \leq (1 - \tau)\left(f(x^0) - f_G\right)$$

Ability to Find Approximate Global Minimizer

(A)POSMM

- Makes rapid progress to f_G
- Outperforms other algorithms *(even while demanding 14-fold concurrency)*



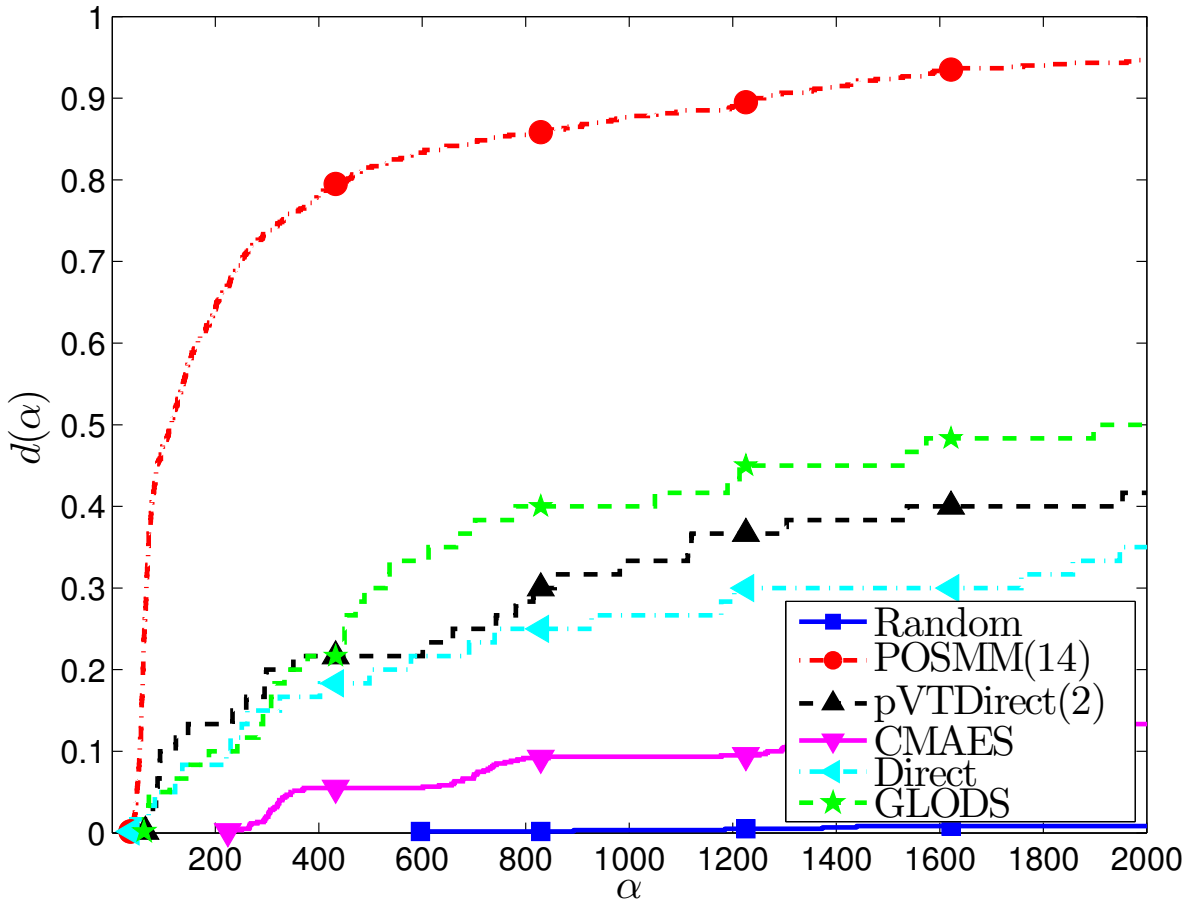
$$\tau = 10^{-5}$$

$$f(x) - f_G \leq (1 - \tau)\left(f(x^0) - f_G\right)$$

Ability to Find j Best Minimizers

(A)POSMM

- Designed to find more than just the global minimizer
- Extends lead for tighter tolerances

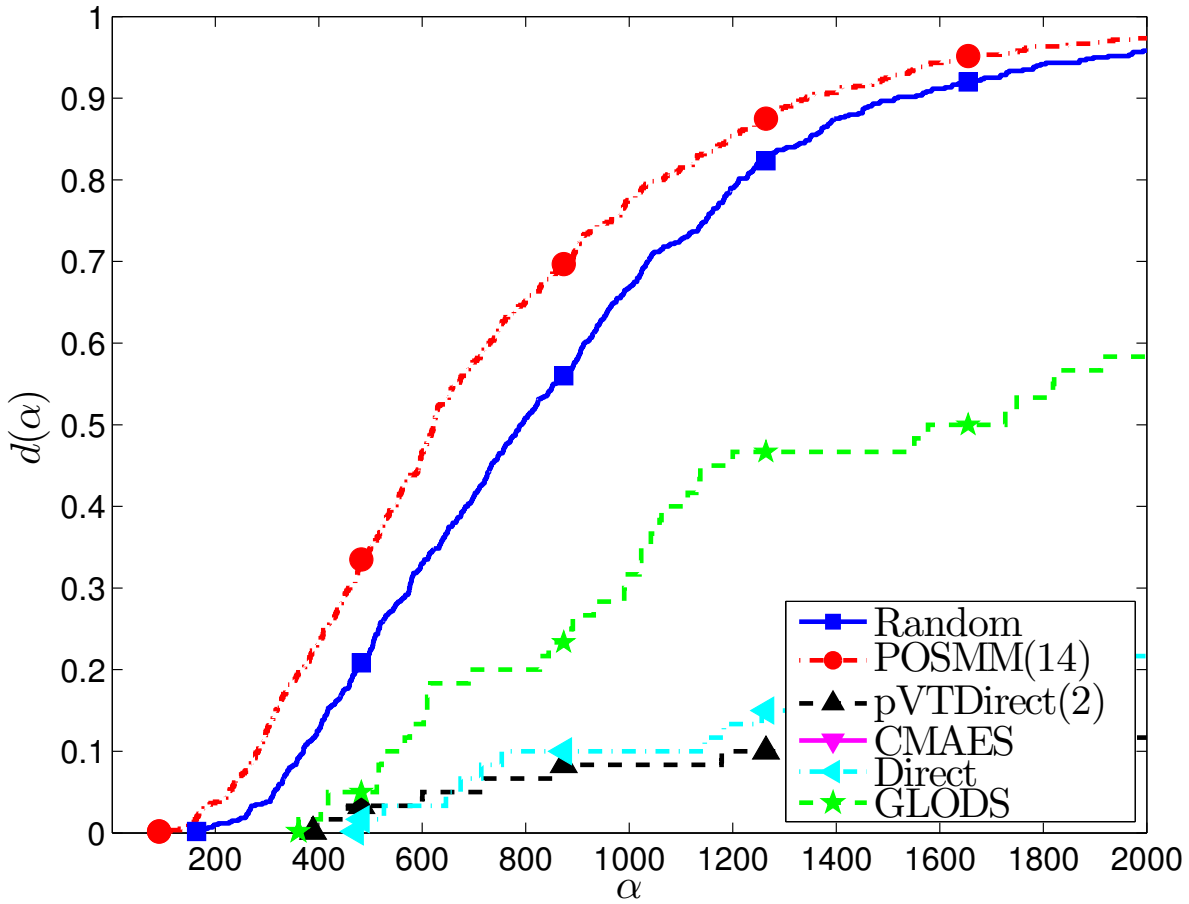


distance $\tau = 10^{-5}$, $j = 2$ minimizers

Ability to Find j Best Minimizers

(A)POSMM

- Designed to find more than just the global minimizer
- Extends lead for tighter tolerances

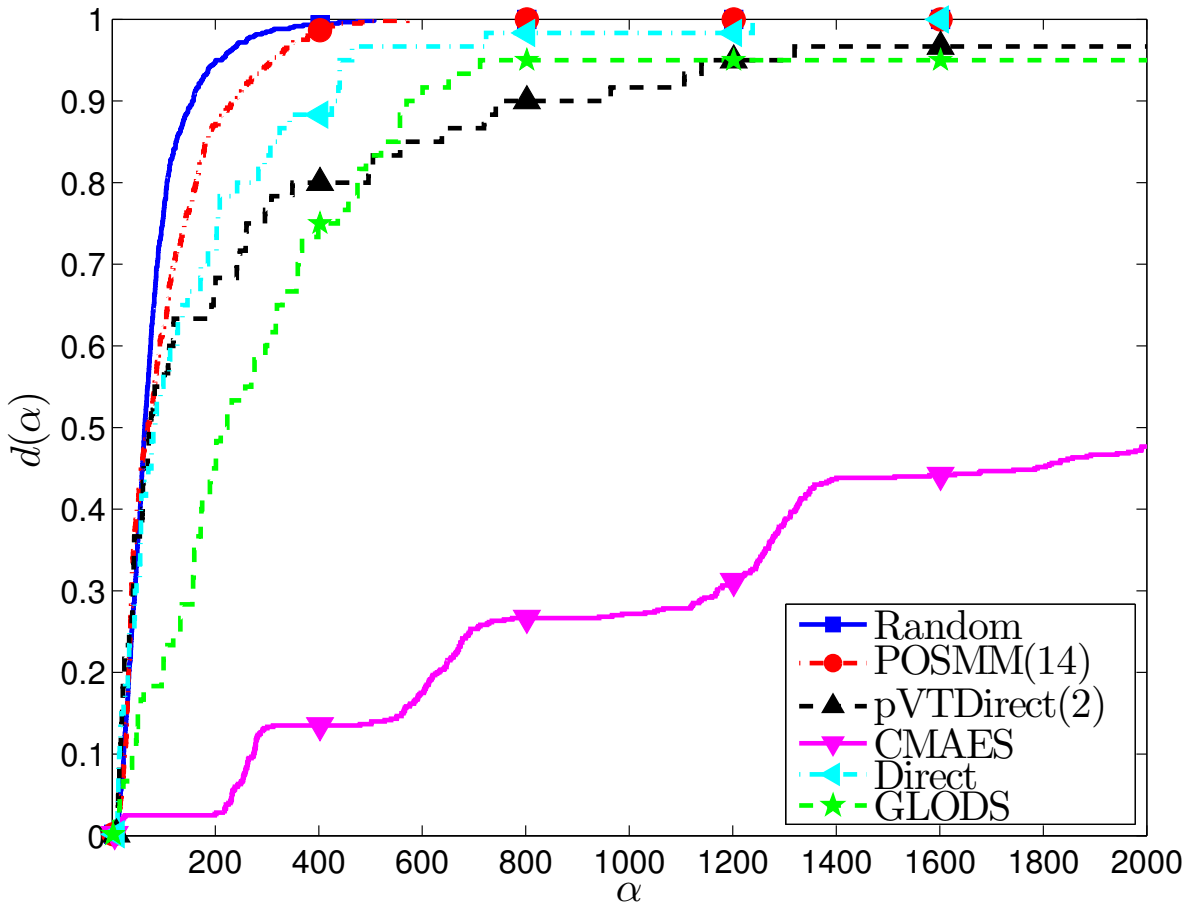


distance $\tau = 10^{-3}$, $j = 7$ minimizers

Ability to Find j Best Minimizers

(A)POSMM

- Designed to find more than just the global minimizer
- Extends lead for tighter tolerances

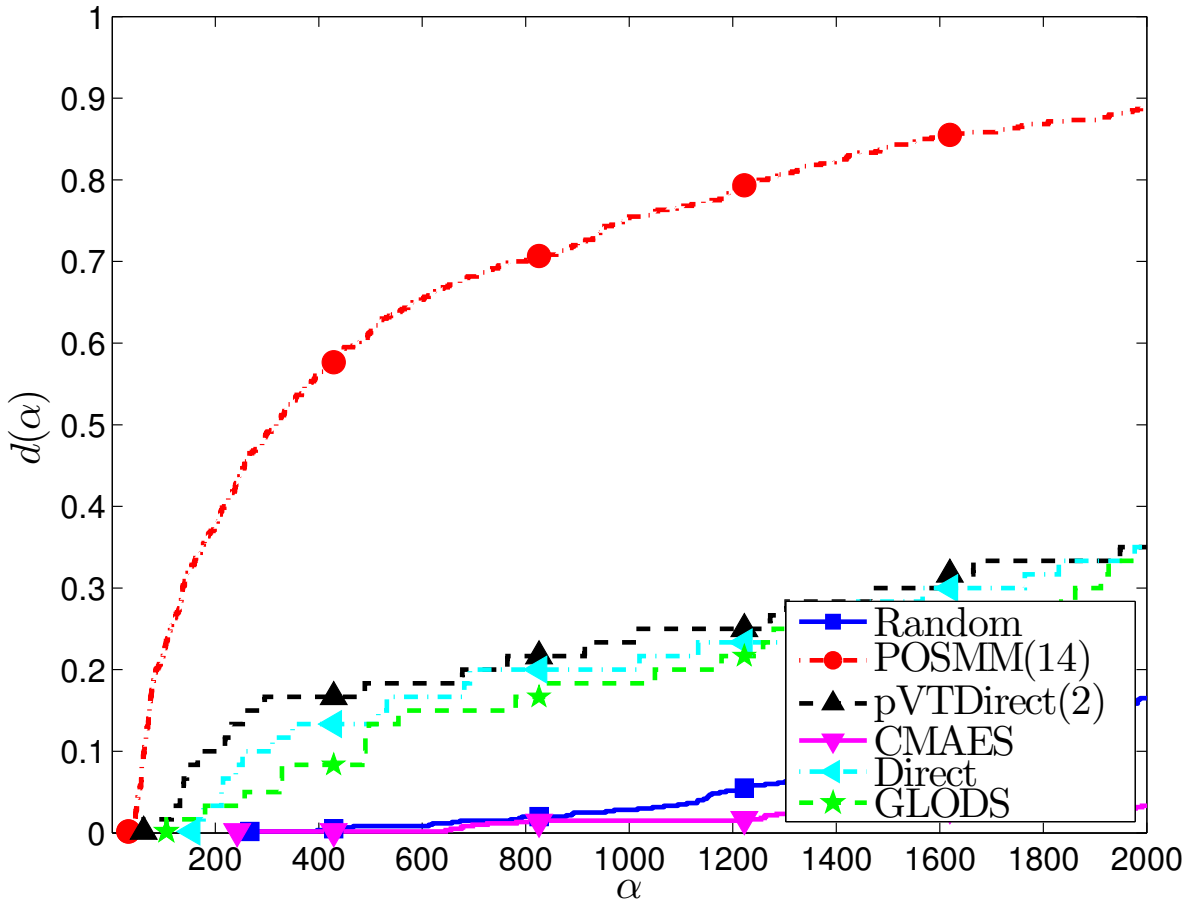


distance $\tau = 10^{-2}$, $j = 3$ minimizers

Ability to Find j Best Minimizers

(A)POSMM

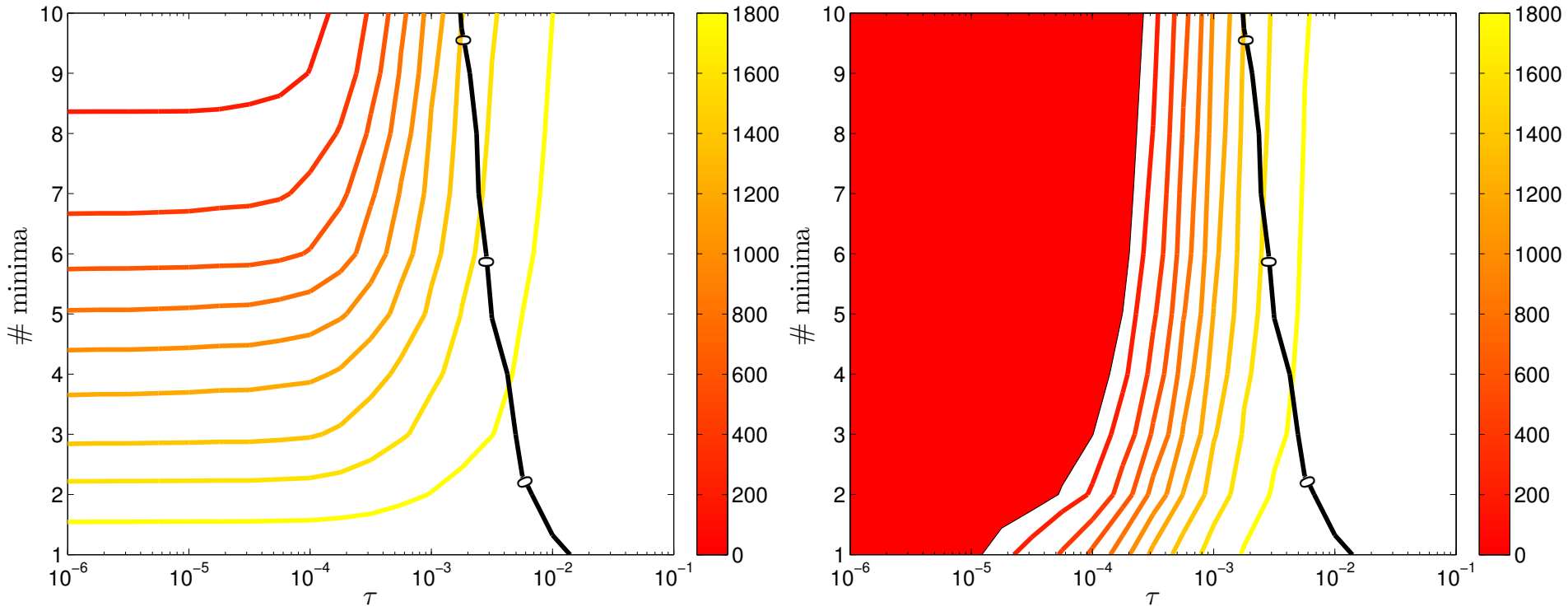
- Designed to find more than just the global minimizer
- Extends lead for tighter tolerances



distance $\tau = 10^{-4}$, $j = 3$ minimizers

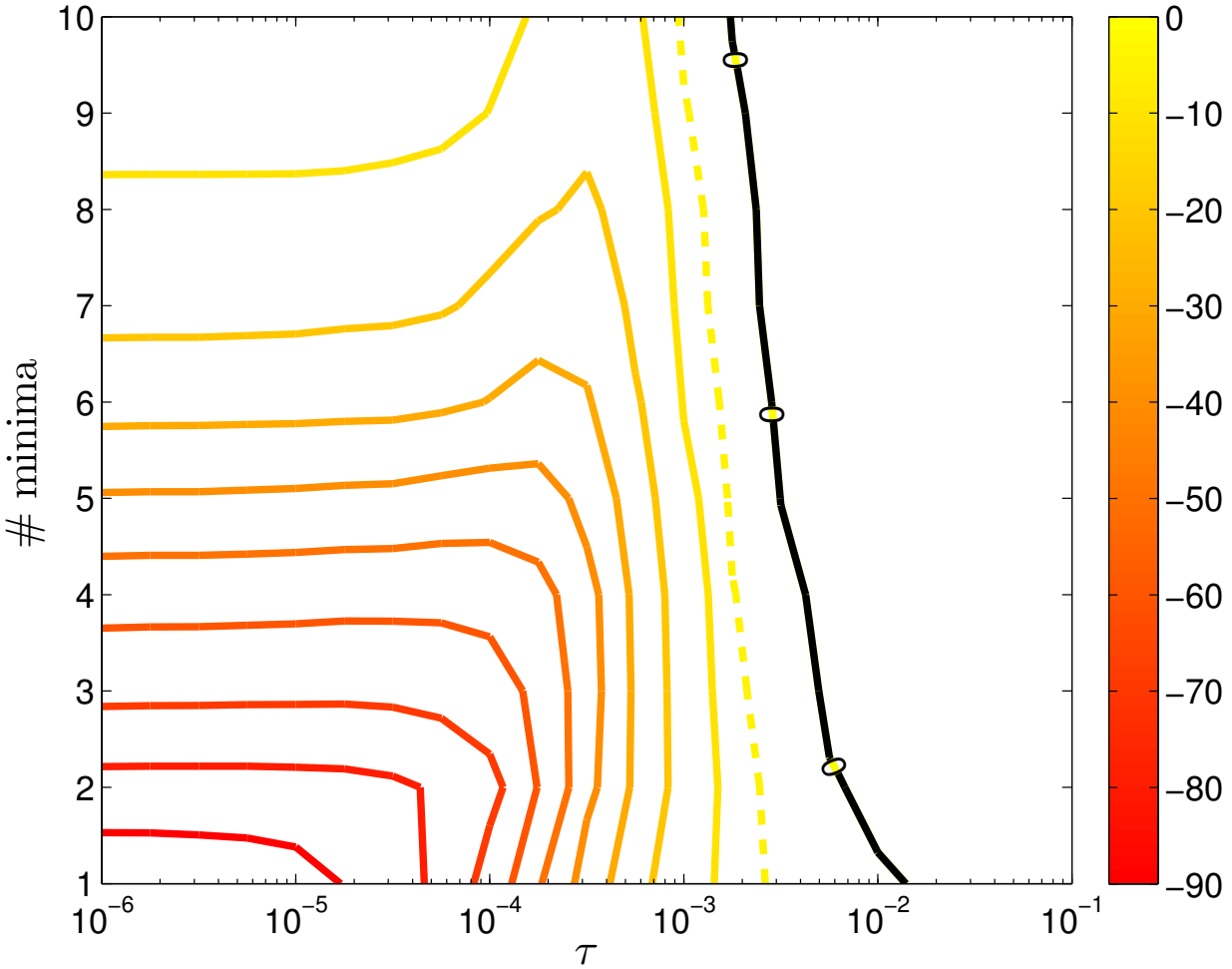
(Tolerance, # Minimizers) Comparison With Random Sampling

Area under data profile: POSMM (L), RS (R)



(Tolerance, # Minimizers) Comparison With Random Sampling

Percent difference between POSMM and RS areas



APOSMM has clear advantages as tolerances tighten

Closing Remarks

- Concurrent function evaluations can locate multiple minima while efficiently finding a global minimum.

Closing Remarks

- Concurrent function evaluations can locate multiple minima while efficiently finding a global minimum.
- Can write/use algorithms that exploit problem structure

Closing Remarks

- Concurrent function evaluations can locate multiple minima while efficiently finding a global minimum.
- Can write/use algorithms that exploit problem structure

Current work:

- Finding (or designing) the best local solver for our framework
- Best way to process the queue?

www.mcs.anl.gov/~jlarson/APOSMM
APOSMM@lists.mcs.anl.gov
jmlarson@anl.gov